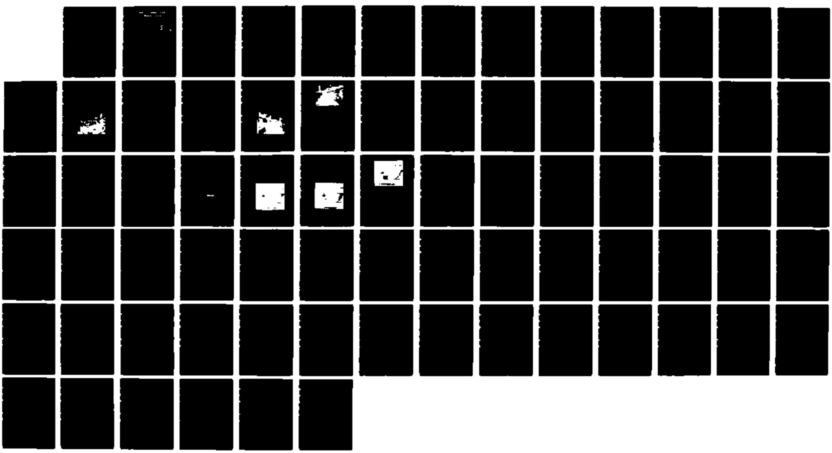
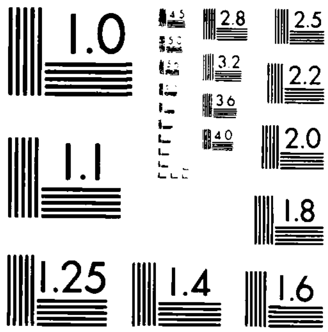
AD-A189 907 UNSTEADY FLOW FIELD MEASUREMENTS USING LDV (LASER DOPPLER VELOCIMETRY)(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA S D HEERICK DEC 87 1/1

UNCLASSIFIED F/G 20/4 NL




1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

DTIC FILE COPY (2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

AD-A189 907

THESIS

UNSTEADY FLOW FIELD MEASUREMENTS
USING LDV

by

Steven Dale Hedrick

December 1987

Thesis Advisor: S. Bodapati

Approved for public release; distribution is unlimited.

88 3 17 031

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION: Naval Postgraduate School; 6b OFFICE SYMBOL (If applicable): 67	7a NAME OF MONITORING ORGANIZATION: Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code): Monterey, California 93943-5000
8a NAME OF FUNDING / SPONSORING ORGANIZATION; 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
UNSTEADY FLOW FIELD MEASUREMENTS USING LDV

12 PERSONAL AUTHOR(S)
Heidrick, Steven D.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM ____ TO ____	1987 December	73

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Laser Doppler velocimetry, schlieren flow visualization, dynamic stall

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The primary goal of this thesis was the development of an experimental technique, and supporting software, for the acquisition and analysis of unsteady velocity data generated by an oscillating airfoil. This research was in support of a major investigation of the compressibility effects on dynamic stall.

The experimental procedure involved schlieren flow visualization for comparison of steady and unsteady flow fields, and for determination of parameters for further study. Laser Doppler velocimetry was employed for obtaining velocity data in the airfoil wake. For unsteady data, the airfoil was oscillated in pitch about its quarter chord.

The data analysis produced wake profile plots representing the flow field disturbed by the airfoil. Results were obtained for steady and unsteady conditions.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT: ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL: Professor Satyanarayana Bodapati	22b TELEPHONE (Include Area Code): (415) 694-4144; 22c OFFICE SYMBOL: 67Bu

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Approved for public release; distribution is unlimited.

Unsteady Flow Field Measurements Using LDV

by

Steven D. Hedrick
Lieutenant, United States Navy
B.E., University of Mississippi, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1987

Author: ______________________
Steven D. Hedrick

Approved by: ______________________
Satyanarayana Bodapati, Thesis Advisor

____
M. F. Platzer, Chairman,
Department of Aeronautical Engineering

____
G. E. Schacher
Dean of Science and Engineering

ABSTRACT

The primary goal of this thesis was the development of an experimental technique, and supporting software, for the acquisition and analysis of unsteady velocity data generated by an oscillating airfoil. This research was in support of a major investigation of the compressibility effects on dynamic stall.

The experimental procedure involved schlieren flow visualization for comparison of steady and unsteady flow fields, and for determination of parameters for further study. Laser Doppler velocimetry was employed for obtaining velocity data in the airfoil wake. For unsteady data, the airfoil was oscillated in pitch about its quarter chord.

The data analysis produced wake profile plots representing the flow field disturbed by the airfoil. Results were obtained for steady and unsteady conditions.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

The research program that provided the opportunity for this thesis was supported by the Army Research Office and the Air Force Office of Scientific Research.

I would like to extend my sincere appreciation to the Fluid Mechanics Laboratory of the NASA Ames Research Center for providing the facilities required for this research. The atmosphere created by its many dedicated professionals did much to stimulate my interests and increase my level of motivation. A special thanks goes to Michael Fidrich for the helpful assistance with the laboratory equipment and, in particular, for the development and construction of the hardware responsible for acquiring airfoil phase data.

I wish to gratefully acknowledge the assistance of Professor Satyanarayana Bodapati, my thesis advisor, for some much welcomed guidance during the course of this work.

I am especially thankful to Dr. M. S. Chandrasekhara, of the Navy-NASA Joint Institute of Aeronautics, for the invaluable assistance provided on this project. His expertise in Laser Doppler Velocimetry and experimental procedures were of tremendous benefit in completion of this research.

Finally, I want to thank my wife, Laura, for her constant support and undying faith in my ability to successfully complete this thesis. Her patience and understanding during

the difficult times did much to sustain my desire for this projec..

I. INTRODUCTION

A. DYNAMIC STALL

The possible benefits and consequences inherent in dynamic stall have stimulated experimental investigations leading to significant progress in understanding this complex phenomenon. Much of the earlier work in this area was carried out by McCroskey using a two-dimensional airfoil section oscillating in pitch about its quarter-chord. The results of this effort succeeded in clarifying the features of dynamic stall up to a Mach number of 0.3 [Ref. 1].

Several significant distinctions between static and dynamic stall should be observed. Static stall is characterized by massive flow separation and loss of lift coinciding with some critical angle of incidence. The aerodynamic forces on the airfoil vary uniquely, or nearly so, with angle of attack. Dynamic stall, occuring during rapid increases in incidence, results in a delay of the associtated loss of lift until well beyond the static stall angle. Dynamic stall also tends to be considerably more persistant. The flow field does not immediately adjust to reductions in incidence angle as with static stall. [Ref. 2]

Dynamic stall is characterized by the shedding and passage of a vortex over the upper surface of an airfoil. The production of this vortex begins shortly after the incidence

has exceeded that of static stall. A thin layer of reversed flow in the boundary layer, evolving into a vortex near the leading edge, represents the initial phase of a dynamic stall cycle. As this disturbance moves across the upper surface of the airfoil, large negative pitching moments, significant drag rise, and a monotonic increase in lift result. The aerodynamic forces, reaching a maximum as the vortex approaches the trailing edge, decrease dramatically with the shedding of this vortex. Reattachment of the flow will then typically occur at some angle less than that corresponding to static stall. [Ref. 2]

B. THESIS GOALS

The research project generating this thesis represents a small part of a major investigation aimed at developing a thorough understanding of the effects of compressibility and pitch oscillation on dynamic stall. The primary goal of the research associated with this thesis was the development of an experimental procedure, and supporting software, for the acquisition and analysis of unsteady flows with a laser doppler velocimetry system. The procedure and software developed was to provide an initial analysis of possible areas of interest and a base upon which to build as the sophistication of the experimental methods increased. A successful attainment of this goal required obtaining velocity data in the wake of an oscillating airfoil and wake profile plots produced from this data.

II. EXPERIMENTAL CONSIDERATIONS

A. EXPERIMENTAL METHODS

1. Laser Doppler Velocimetry

Since its introduction in 1964, laser Doppler velocimetry has proven to be a highly reliable means for obtaining measurements in complex turbulent flows. The method is capable of performing instantaneous, non-intrusive velocity measurements under a wide range of conditions. Being relatively independent of fluid properties, accurate measurements are possible without the requirement of calibration. [Ref. 3]

The basic principle behind laser Doppler velocimetry is the measurement of the velocity of particles travelling with the fluid. These particles, when illuminated with a focused laser beam, become sources of scattered light. The Doppler shift of the scattered light is then used to determine the velocity of the particles and, depending upon how well the particles are following the flow, the fluid velocity as well.

As a minimum, six components are necessary for a complete LDV system: laser source, transmitting optics, receiving optics, photodetector, signal processor, and data processor. The function of the transmitting and receiving optics are focusing and light collecting, respectively. The photodetector serves to convert light signal to an electrical

signal which is then used by the signal processor to provide a measurement of the frequency. The capability of detecting reversed flows requires an additional component for frequency shifting.

Due to the wide range of disciplines involved, optics, electronics, light scattering, and signal processing, the implementation of this method can be difficult [Ref. 3]. Consequently, the expertise of the operator can have a profound impact on the results of LDV measurements. It is therefore imperative, to an even greater degree than many other methods, that considerable attention be given to applying correct measurement technique.

2. Schlieren Flow Visualization

Schlieren flow visualization systems provide a qualitative analysis of the density gradient in compressible flow fields. The fundamental concept of this method is the variation of the refractive index gradient of a fluid with density.

A basic schlieren system includes a light source, transmitting and receiving mirrors, a knife edge, and a focusing screen. The transmitting mirror serves to direct a parallel light beam through the test section and onto the receiving mirror. An image of the light source is then formed at the focal point of this mirror. The knife edge, placed in the plane of the light source image, cuts off part of the transmitted light and controls the contrast of the image focused on the screen. Changes in test section density,

therefore, produce variations of light intensity and provide a visual representation of the flow field. [Ref. 4]

B. INSTRUMENTATION AND EQUIPMENT

1. Equipment Setup

The experimental procedures for this research were conducted at the NASA Ames Fluid Mechanics Laboratory. A 25-cm by 13-cm indraft wind tunnel with a slotted wall test section was used. Figure 2.1 shows the test section with plenum chambers above and below the slotted walls. The adaptive wall design of this test section was developed for use in previous experiments. Although the technique was not

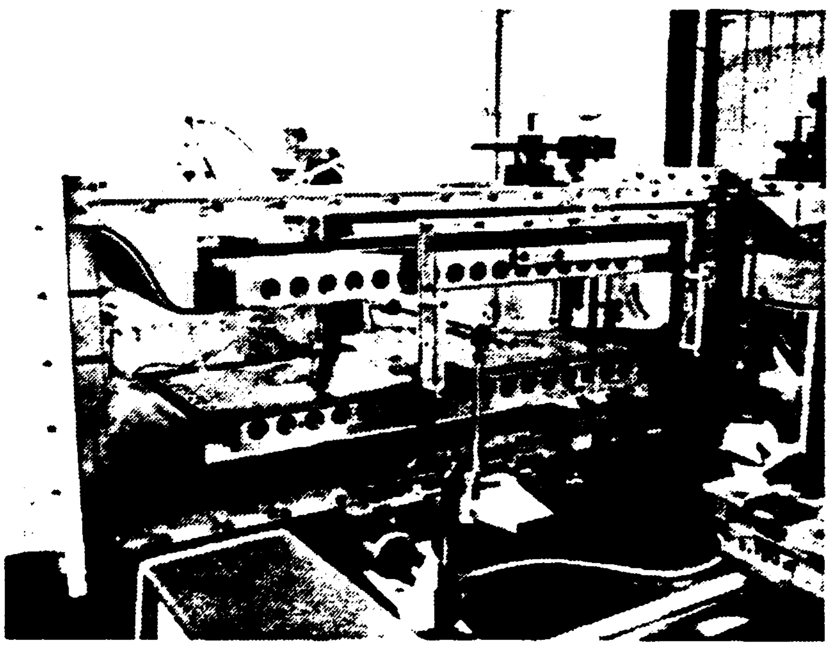


Figure 2.1 Wind tunnel test section.

used, the slotted walls were helpful in reducing wall interference.

The model for the experiment was a NACA 0012 airfoil with 3" chord. The airfoil, attached to the test section walls at quarter chord, was connected with mechanical linkage to an electric motor, as shown in Figure 2.1. This variable speed motor provided a means for oscillating the model about a pitch axis and obtaining unsteady data at various frequencies.

An optical encoder, also driven by the motor, provided 3,600 counts per revolution of 14-bit data on the phase angle of the oscillating airfoil. The characteristics of the encoder were absolute in that a whole word output, with a unique code pattern, represented each of the 3,600 discrete positions. This code was derived from independent tracks on the encoder disk corresponding to individual photodetectors.

Test section Mach number and velocity were obtained by using a 24-port Scanivalve with a differential pressure transducer. Calculation of these values required measurements from four of the ports, two for Mach number computation and two for calibration of the transducer. The linear relationship between the measured pressure and the corresponding voltage produced by the transducer was used for determining the pressures. In accomplishing this, one port of the Scanivalve, vented to the atmosphere, measured ambient pressure and a second port, tapped to a mechanical vacuum pump, measured a calibration pressure. A conversion factor

was then established by dividing the difference of the known pressures, atmospheric and calibration, by the difference in voltages produced by the transducer.

Location of the LDV system probe volume, or beam crossing point, was controlled by a three axis traversing mechanism. A Summit Taskmaster numerical control computer, originally developed for controlling movements of industrial machine tools, was used for this task. The numerical control computer (NCC) was capable of manual operation, programmed operation, and, with software adaptation, operation from the computer workstation. The NCC provided control inputs to two sets of electric motors, one for the transmitting optics and the other for the receiving optics. The necessity of using different traversing motors for the transmitting and receiving optics placed a great demand on the synchronization of this system. Due to the high sensitivity of the LDV optics train, the synchronized movement was often inadequate for maintaining signal strength.

Phase encoder data and digitized velocity data from the LDV signal processors were channeled to a 3D-LDV Computer Interface (CI). The CI, multiplexing the inputs to a single digital data channel output, provided 16-bit parallel output data to the computer data bus. The CI was capable of accepting up to six channels of digital data and from one to three event pulses from the signal processors. The multiplexing cycle could be initiated by random event pulse inputs or synchronized inputs, the latter being necessary for unsteady

data acquisition. When operating in the synchronized mode, two or three pulses, as selected, occuring within a set time (coincidence time) would be required to initiate the multiplex cycle and feed data to the computer.

2. Laser Doppler Velocimeter Subsystems

A two component LDV system was used to obtain the flow field velocity data. Figures 2.2 and 2.3 show the general arrangement of the system components for the transmitting and receiving optics sides, respectively.

A 5 Watt Argon ion laser, Spectra Physics Model 164, was used for the system. The unit was operated in the

Figure 2.2 LDV transmitting optics.

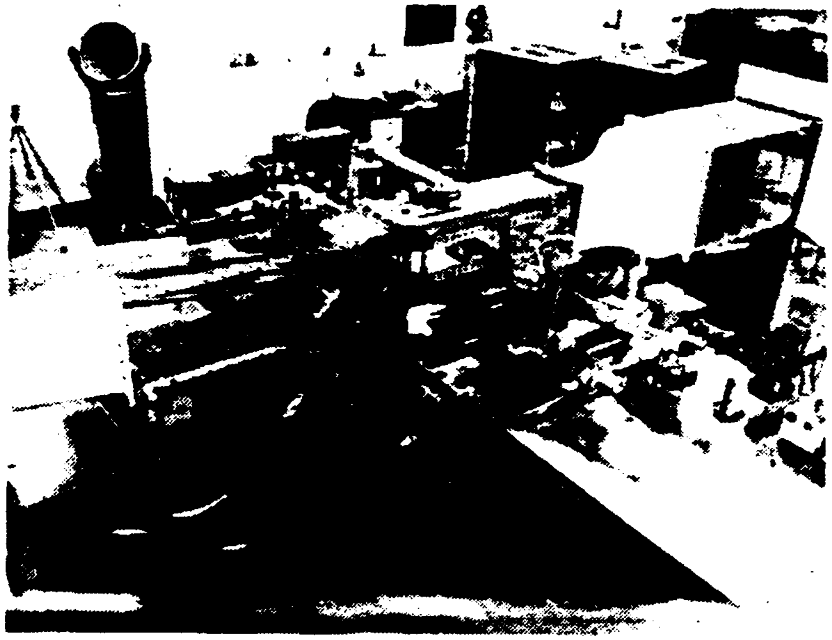


Figure 2.3 LDV receiving optics.

multiline mode and produced a vertically polarized beam composed of all wavelengths inherent to this type of laser.

Figure 2.4 illustrates the sequence of steps required of the transmitting optics. A set of mirrors was first used to lower the multiline beam to the level of the optical components. After passage through a collimator, used to ensure collimation of the beam and subsequent crossing of the beams at their waists, the beams pass through an attenuator. The function of this component was to reduce the beam intensity without inhibiting transmission of any wavelengths, a possible consequence of simply reducing the power output of the laser. Prior to separation of the wavelengths by the dispersion prism, it was necessary to horizontally polarize

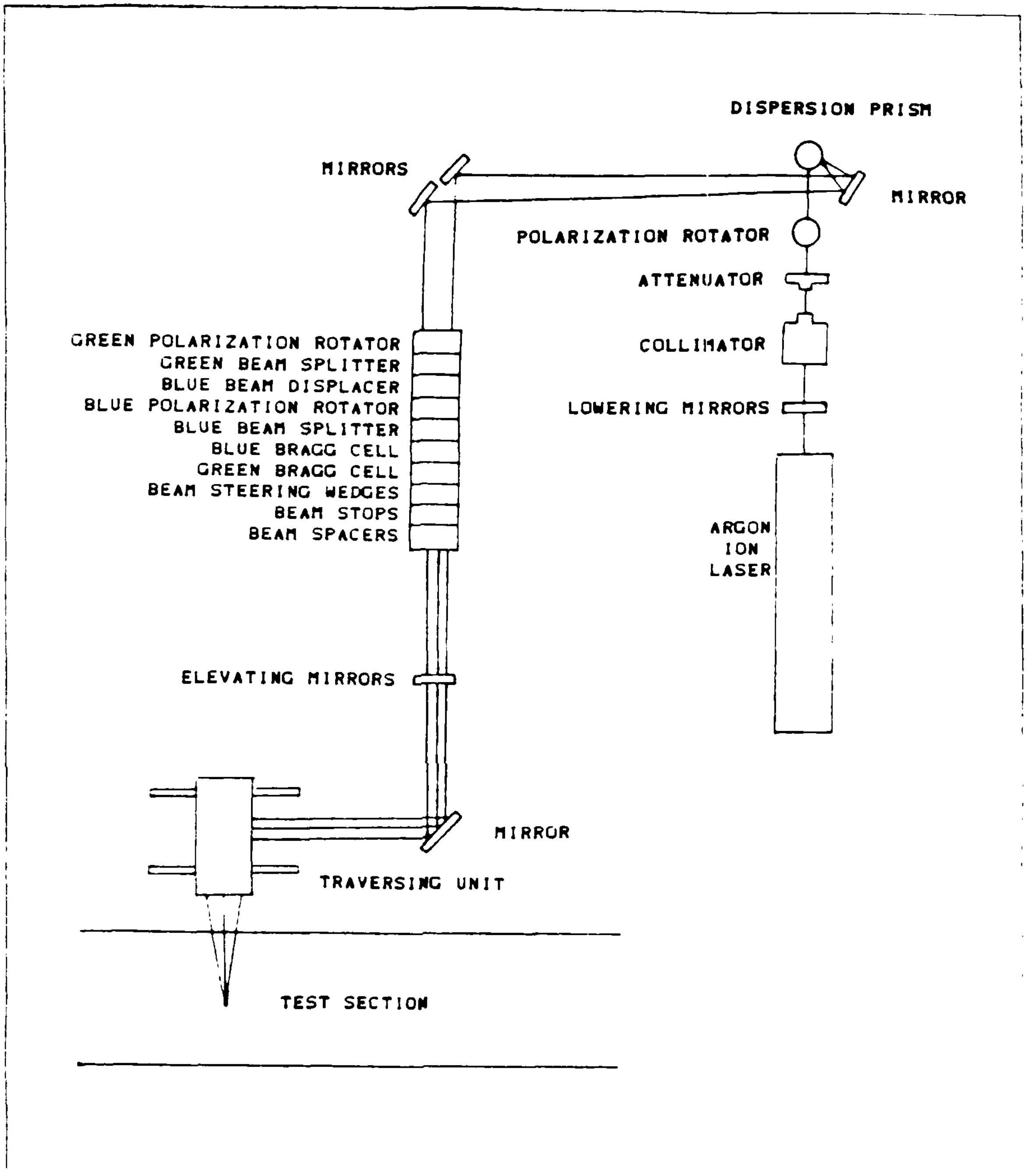


Figure 2.4 Two-component transmitting optics.

the beam by passing it through a half-wave plate, or polarization rotator, in order to obtain maximum light transmission. The two stronger wavelengths, 514.5 nm (green) and 488 nm (blue), were then guided by turning mirrors to the remaining optical components.

Before reaching their respective Bragg Cells, the green beam was split vertically and the blue was displaced to the center of the component assembly, vertically polarized, and split horizontally. The relationship between the splitting and polarizing planes was driven by signal efficiency. High signal-to-noise ratio requires split beams to have equal intensities. The beams will be of approximately equal intensity when the polarity of the incident beam is perpendicular to the plane of the exiting beams [Ref. 3]. Therefore, the polarity of the vertically split beam was maintained as horizontal and that of the horizontally split beam was rotated to vertical. As later discussed, this difference in polarity was also necessary for separation of the two wavelengths by the polarization beamsplitter in the receiving optics.

Bragg Cells are acousto-optic devices used to generate a frequency shift in the beam and allow for resolution of the 180 degree directional ambiguity. One beam of each of the two wavelengths was directed through a Bragg Cell producing a 40 MHz shift. The unshifted beams were passed through glass rods to compensate for the path length difference experienced by the shifted beam. Due to the nature of

the Bragg Cell, the frequency-shifted beam was deflected from its initial alignment. Therefore, beam alignment prisms, or steering wedges, within the Bragg Cell modules served to bring the beams back to parallel.

A microscope objective, placed in the test section at the probe volume, was used to observe the beams as adjustments were made to ensure the beams focused to a common point. If the two probe volumes are not at the same location within the test section, then the measurements will be made at different points. The necessity for all four beams to overlap at the focal point was especially prevalent for the acquisition of unsteady data due to the need for a high coincidence data rate.

Receiving optics are designed to image, to the greatest extent possible, the light scattered at the probe volume onto the photodetectors. Since the scattered light intensity is the greatest in the forward direction, a forward scatter system is most often preferred for obtaining the greatest signal-to-noise ratio. The system used for this analysis was an off-axis, forward scatter configuration. Off-axis collection angles usually allow good noise isolation by admitting only light scattered at the sample volume and improve measurement resolution by reducing the probe volume length. [Ref. 2]

The first three lenses in the receiving optics were used to collect scattered light from the probe volume and collimate the beam (Figure 2.5). The beam was then directed

to a polarization beamsplitter for separation of the two colors. The two components were then passed through line filters, for blocking ambient light and preventing cross talk, and to lenses used for focusing the beams onto the pin-hole aperature of their respective photomultiplier tubes. The photomultiplier tubes converted the light energy into electronic signals which were then passed to signal processing equipment. Preamplifiers were located in close proximity to the photomultiplier tubes and used to increase the signal level before the introduction of additional noise into the cables.

Signal processing was accomplished with Macrodyne counter processors. Counter processors operate by employing a crystal controlled oscillator to time over a selected number of cycles within the burst signal. The clock count is initiated when the filtered signal exceeds a set threshold. The Doppler frequency is then determined by the time required to count a specified number of cycles. The Macrodyne processors use a 125 MHz clock and count over 8 or 16 cycles, as selected.

The Macrodyne processors use a periodicity check coupled with a three-level validation scheme to reject noise. The periodicity check uses two counters to make period measurements over a different number of cycles, 5 and 8 in this procedure. If the ratio of the two measurements is not within a specified percentage of 5/8, the signal is rejected. Three-level validation requires a signal to exceed a positive

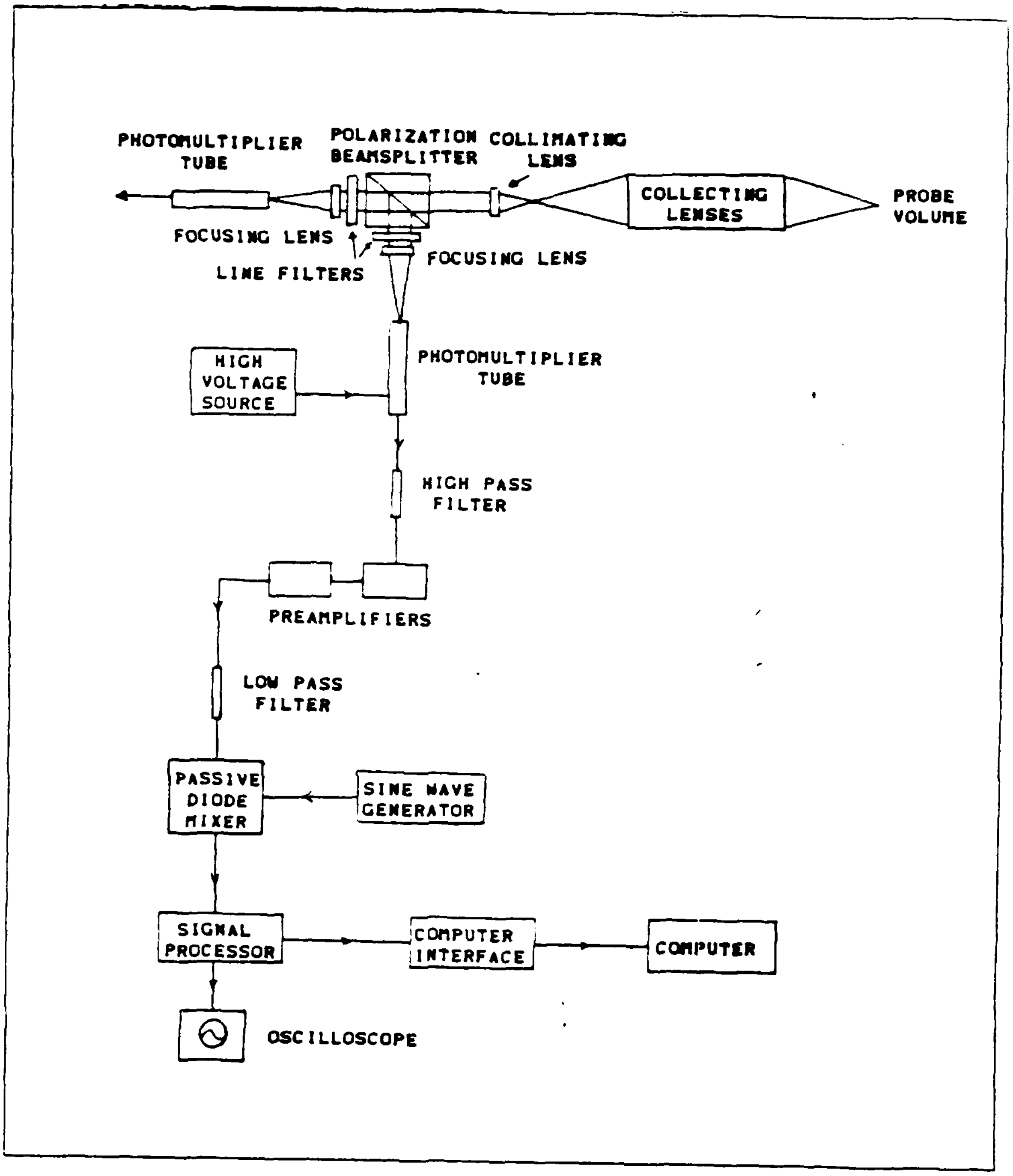


Figure 2.5 Data collection and processing.

threshold level, pass through zero, and then exceed a negative threshold level before the next zero crossing. If these conditions are not met on any cycle the counter will reset and immediately process a new Doppler burst. This method prevents rejection of potentially good signals having noisy cycles at the beginning of the burst.

3. Schlieren System

The Schlieren system included two concave mirrors, each with a 120 inch focal length, a stroboscopic light source, a vertical knife edge, a focusing lens, and a polaroid film support for obtaining photographs of the flow field. The stroboscope, a type 3015 Strobrite, has a maximum output intensity of 300 million beam candles and, using an internal oscillator, can provide flashing rates in excess of 1,000 per second. By adapting a lens cover with a vertical slit to the stroboscope a slit source was obtained and the resolution of the image significantly improved.

To obtain the unsteady photographs a digital comparator was used to trigger the stroboscope and provide the light for the photographic imaging on the polaroid film. The comparator received 14-bit phase input from the optical encoder and provided the triggering pulse when the instantaneous phase angle of the airfoil coincided with the angle specified to the comparator.

C. DATA ACQUISITION AUTOMATION

A Digital VAXstation II minicomputer with graphics workstation was used to automate the data acquisition process. The signal path is illustrated in Figure 2.5 starting from the probe volume, through the signal processing phases, and terminating within computer memory. The main program used in accomplishing this task, 'LDV_ACQ', was originally developed to read data from the analog to digital converter. This was later modified to obtain pressure data for calculation of the test section velocity and call supporting subroutines for the acquisition and analysis of the velocity data.

The subroutines assisting with various aspects of the acquisition process are listed below and included in the appendix of this report.

(1) HWTRVT - allows control of the traversing mechanism from the computer terminal by sending character data to the interface.

(2) HWWA - combined with an internally called subroutine, DAMUX, retrieves data from multiplexer, interprets phase and velocity information, creates data files, and generates histogram arrays and scaling parameters.

(3) PT_PLOT - provides graphical display of computed velocity values corresponding to airfoil phase angles.

(4) PPLOT - similar to PT_PLOT but provides the additional capability of allowing the user to vary the plotted velocity range.

(5) HIST - plots a histogram of the percentage of data points which fall within specified velocity limits.

III. DATA ACQUISITION

A. TEST SECTION VELOCITY

The main program for the acquisition process, 'LDV_ACQ', was responsible for the calculation of test section velocity. In combination with scanivalve positioning subroutines, the program acquired pressure transducer data from 7 ports. As previously discussed, measurements from 2 of the ports were used for calibration. Pressure data from 3 ports was discarded, and the other two provided the static pressure at the entrance to the test section and the total pressure measured at the nozzle inlet. Assuming isentropic, perfect gas conditions, this supplied the necessary parameters for the Mach number computation by employing the following relationship:

$$M = (((P_0/P)^{(\gamma-1)/\gamma}-1)*(2/(\gamma-1)))^{1/2}$$

with a value of 1.4 used for γ. The test section velocity was then calculated by:

$$V = M \times a$$

where a, the sonic velocity, was obtained as follows:

$$a = \sqrt{\gamma RT}$$

The temperature required for this calculation was read from a wall thermometer in the test cell and entered when prompted by the program.

The velocity values were transferred to data files and later used to normalize the LDV velocity data.

B. SCHLIEREN PHOTOGRAPHY

A qualitative analysis of the steady and unsteady flow fields was conducted at two Mach numbers, 0.3 and 0.5, and angles of incidence varying from 0 to 20 degrees. In accomplishing this, test section Mach number was set by adjusting the wind tunnel throat and verified with Scanivalve pressures and calculations from 'LDV_ACQ'. Final adjustments to the Stoboscope intensity and knife edge were then made to ensure optimum contrast.

C. WAKE VELOCITIES

With the desired wind tunnel conditions established and the probe volume in position for the data point, 'HWWA', a subroutine called by the main program, was employed for raw data acquisition and conversion to velocity and phase information. An internally called subroutine, 'DAMUX', acquired 4 or 6 channels of data, as selected, from the LDV Computer Interface (CI). The CI was capable of transferring either 2, 4, 6, or 8 channels of data, with the first two of the channels reserved for time and status information generated by the CI. During the acquisition of steady state velocities, 4 channels were selected to provide the 2 velocity components in addition to time and status data, which were discarded since they were not relevant for the experiment. For unsteady acquisition, 6 channels of data were transferred to accommodate the additional data channel required for phase input.

As previously mentioned, the signal processors determined the Doppler frequency by using the time required to count a specified number of cyles, 8 for this analysis. This raw data was transferred to the computer in 16 bits, with a 10 bit mantissa, a 4 bit exponent, and 2 nonessential bits used for controlling data flow. Due to the format of the time words, it was necessary to invert and reverse the bits before extracting the exponent and mantissa values. The signal period was then calculated as follows:

$$ts=(1/32)(D9D8...D0) \times 2$$

where D9D8...D0 represented the binary mantissa and EXP was the range selected with the processor, used to divide the number of clock counts in the Doppler burst.

The signal frequency, the inverse of the period, was obtained from:

$$f_s = (32 \times 1000)/(2^{EXP} \times \text{mantissa})$$

The signal frequency was composed of the Doppler frequency, the 40 MHz shift frequency, and the downmixing frequency. This allowed the Doppler frequency to be deduced as follows:

$$f_d=f_s-f_{shift}+f_{mix}$$

The 8 cycles counted within the Doppler signal physically represent 8 fringes crossed by a particle travelling with the flow. Consequently, the velocity component perpendicular to the fringe pattern may be calculated by the product of the Doppler frequency and the fringe spacing. The equation for the latter is:

$$\delta = \lambda / [2 \cdot \sin(\theta/2)]$$

where λ is the wavelength of the beam and θ is the angle between the intersecting beams.

After obtaining a sufficient number of data points at a probe position (2,000 for unsteady flows), the velocity and phase data, as well as the probe position, test section velocity, and mixing frequencies, were stored in files for later analysis. The Taskmaster traverse controller was then used to move the probe volume and data collection continued.

The data acquisition software included two routines for on-line performance checks of the data. 'HIST' was used to provide a histogram of the percentage of data points which occurred within specified velocity limits. Figure 3.1 shows a sample plot provided by the subroutine. This data validation method is most useful for steady flow conditions, in which case a high percentage of data concentrated around the estimated velocity would be a positive indicator. For unsteady analysis 'PT_PLOT' provided a graph of computed velocity values corresponding to airfoil phase angles. With this information available, the data quality could be easily assessed and a determination made regarding the acceptance of these values.

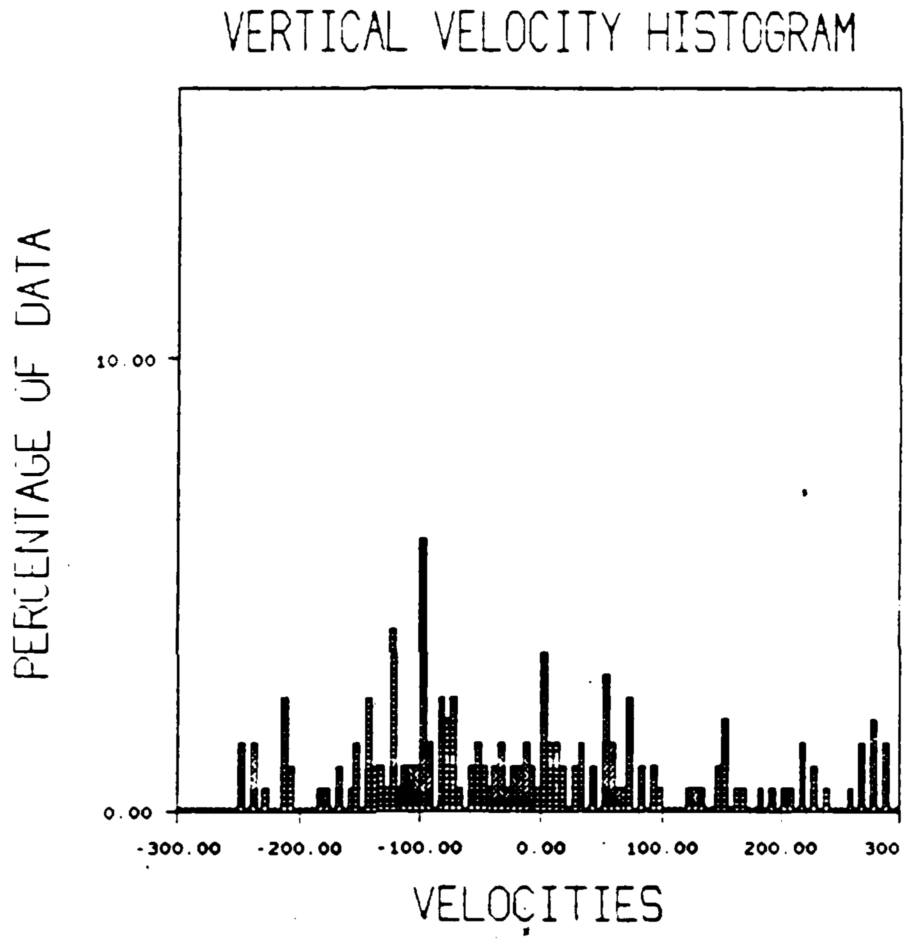


Figure 3.1 Sample histogram.

IV. RESULTS AND DISCUSSION

A. SCHLIEREN PHOTOGRAPHS

The Schlieren photographs obtained from the analysis of the steady flow field were intended to provide information on the parameters corresponding to early flow separation and a reference for comparison with unsteady flows at the same conditions. Steady state separation was first observed at a Mach number of 0.5 and a 10 degree angle of incidence (Figure 4.1). Based on this, the parameters for the unsteady

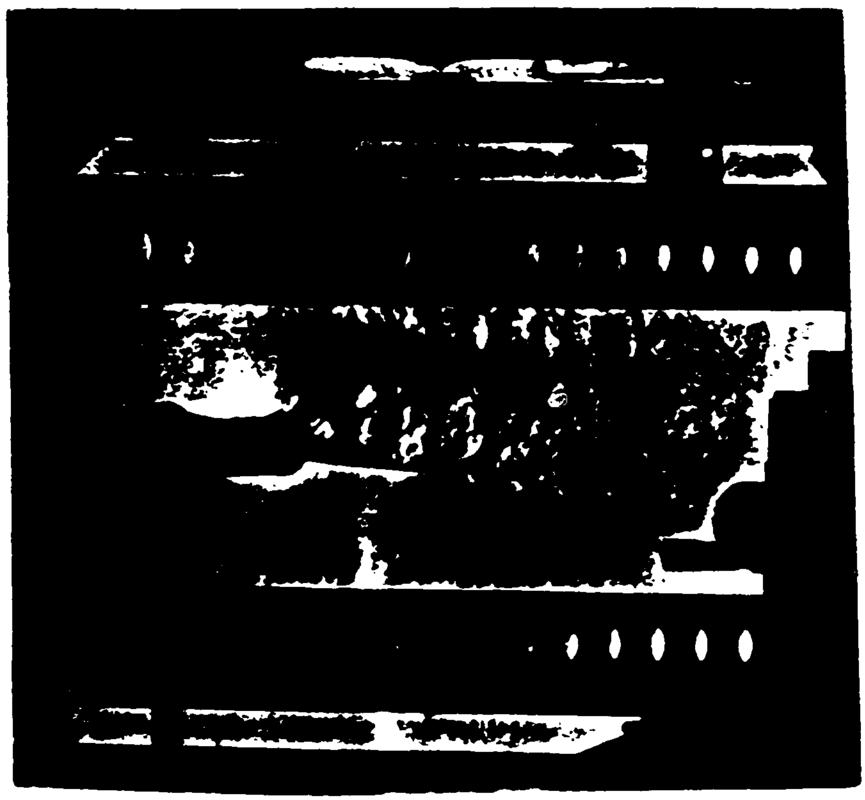


Figure 4.1 Steady state Schlieren photograph; mean α =10 degrees.

analysis were selected to be a Mach number of 0.5 and a phase angle corresponding to a 10 degree angle of incidence. Unsteady flow photographs were then obtained at oscillation frequencies of 5 and 25 Hz for comparison with each other and with wake profiles generated from LDV data (Figures 4.2 and 4.3).

From a comparison of the steady and unsteady photographs, it would appear that oscillating the airfoil reduces the size of the separated wake and, consequently, mitigates the effect of stalling. This is more apparent at the higher frequency

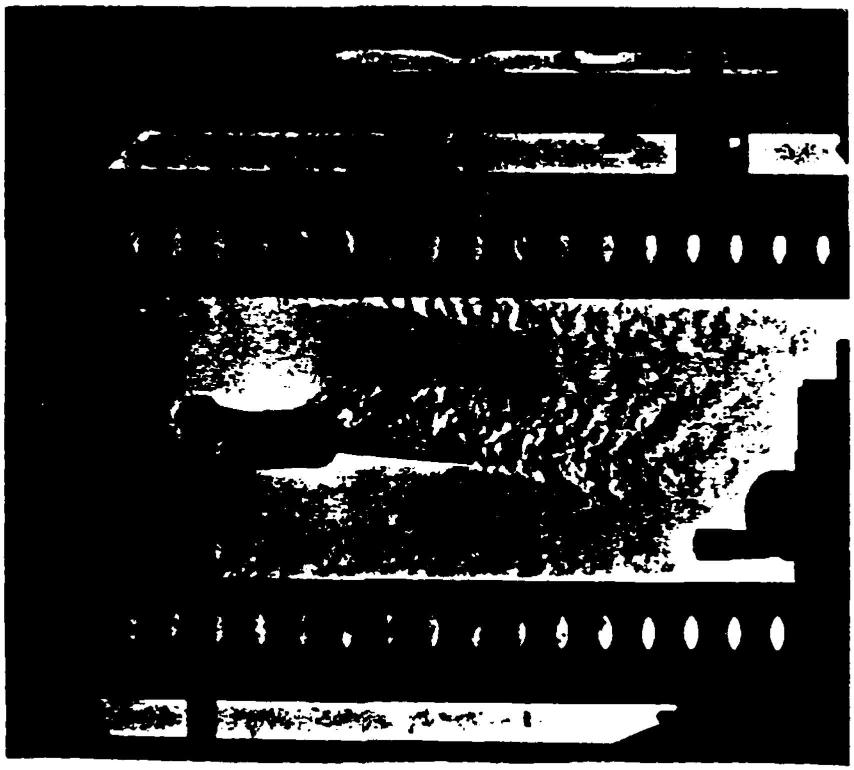


Figure 4.2 Unsteady flow Schlieren photograph; $f=5$ Hz, $\alpha=10$ degrees.

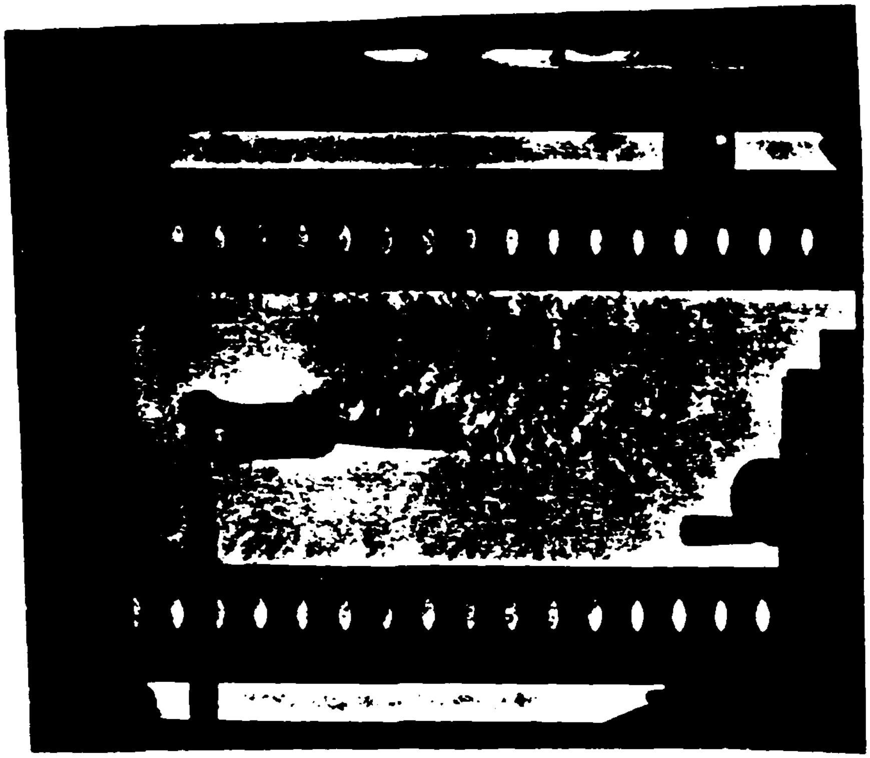


Figure 4.3 Unsteady flow Schlieren photograph; $f=25$ Hz, $\alpha=10$ degrees.

oscillation, a logical result due to the greater effect of time unsteadiness at higher reduced frequencies.

B. DATA ANALYSIS SOFTWARE

The main program, 'ORGIZE', and its supporting subroutines were designed to provide wake profile plots from the data files generated during acquisition. To accomplish this, the data file corresponding to each point in the profile is input to the software. A series of routines first reorganizes the data files and discards statistically bad data points, those outside of two standard deviations from the

mean. Due to the oscillatory nature of the forcing function producing the unsteady data, a sinusoidal curve fit is then introduced to allow evaluation of the first harmonic of the velocity distribution over the phase. A unique sinusoid is generated for each data point within the profile.

A minor degree of uncertainty is introduced into the velocities by the data interpolation and curve fitting routines discussed above. However, unlike many other velocity measurement techniques, LDV data is obtained randomly due to the random arrival of the seed particles at the prove volume. Without this curve fit, it would be difficult to generate velocity information at the same desired phase angle for each data point in the profile.

During the generation of the wake profile the user is provided a comparison plot of mean values and the sinusoid curve fit. The mean values are provided by 'BDPTS' and computed within each 20 degree phase window, as illustrated in Figure 4.4. This was helpful in ascertaining the quality of the curve fit and, consequently, the quality of the wake profile results.

The data analysis routines are summarized below and included in the appendix of this report.

(1) ORGIZE - main program, calls supporting subroutines; sequentially organizes data to assist with subsequent analysis.

(2) PT_PLOT - provides graph of computed velocity values corresponding to phase.

(3) BDPTS - computes mean and standard deviation of velocities occurring within 20 degree phase windows;

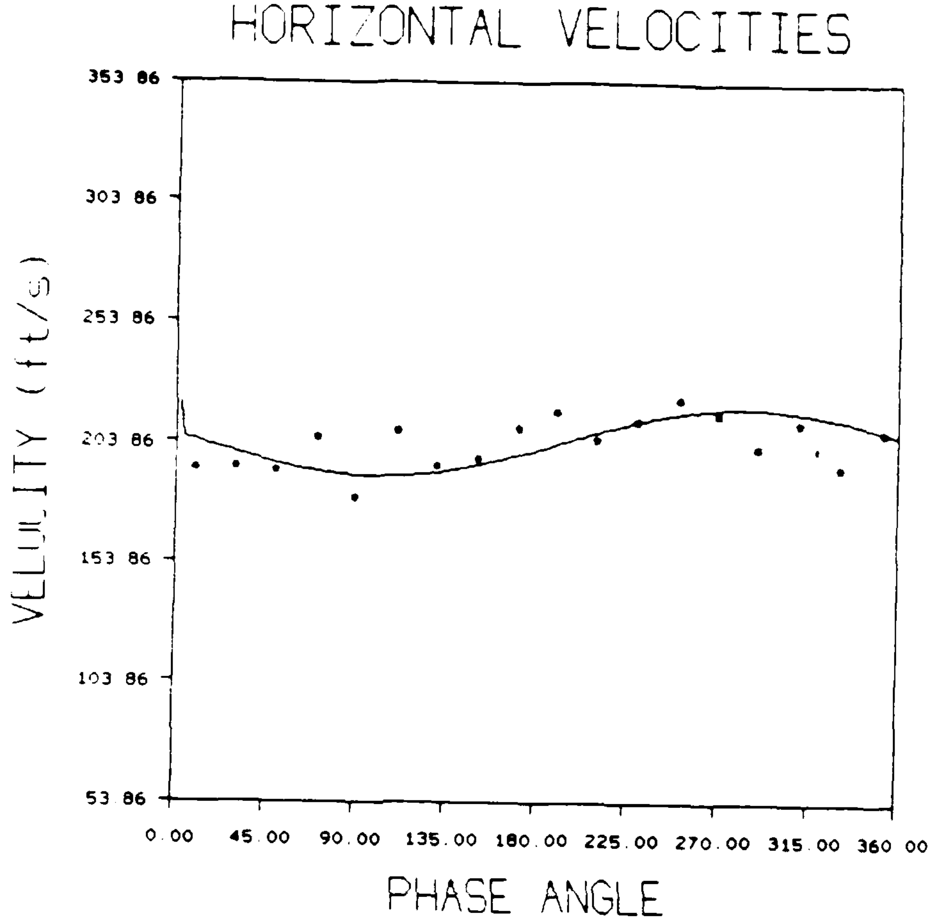


Figure 4.4 Comparison of mean and curve fit.

discards data points outside of two standard deviations of the mean within each of these windows; provides standard deviation plots of data.

(4) AVG - averages data values occurring at same discrete phase.

(5) INTERP - provides a linear interpolation between data, resulting in 180 equally spaced points; necessary for input to FOURIER.

(6) FOURIER - computes coefficients for a sinusoidal curve fit of the data (adapted from June 1977 OCS Airfoil program).

(7) CVFIT - fits sinusoidal curve to velocity data (utilizes FOURIER).

(8) PLOT - provides graphical comparison of curve fit with mean values generated by BDPTS.

(9) PROPLOT - provides wake profile plots.

C. WAKE PROFILES

LDV data was obtained for steady and unsteady flows at a Mach number of 0.5 and a 10 degree angle of incidence. The steady profiles were collected at distances of 0.5, 1.0, and 2.0 inches behind the trailing edge of the airfoil. These values were non-dimensionalized by the chord length of the airfoil and are indicated as such on the profile plots. Due to time limitations, only one data profile was collected for the unsteady case, at two inches behind the trailing edge.

The wake profile plots are presented in the following pages of this section. A lack of data points in the lower part of the flow field will be apparent upon inspection of these results. This was due to a mechanical restriction on the traversing mechanism.

The curve fit of the plotted values of the steady data was approximated and hand drawn to aid the reader in observing the effects of the airfoil in the flow field. The software provides a spline interpolation fit between the data, as used in the unsteady plots. However, due to the amount of data scatter involved, this method was not used for these results.

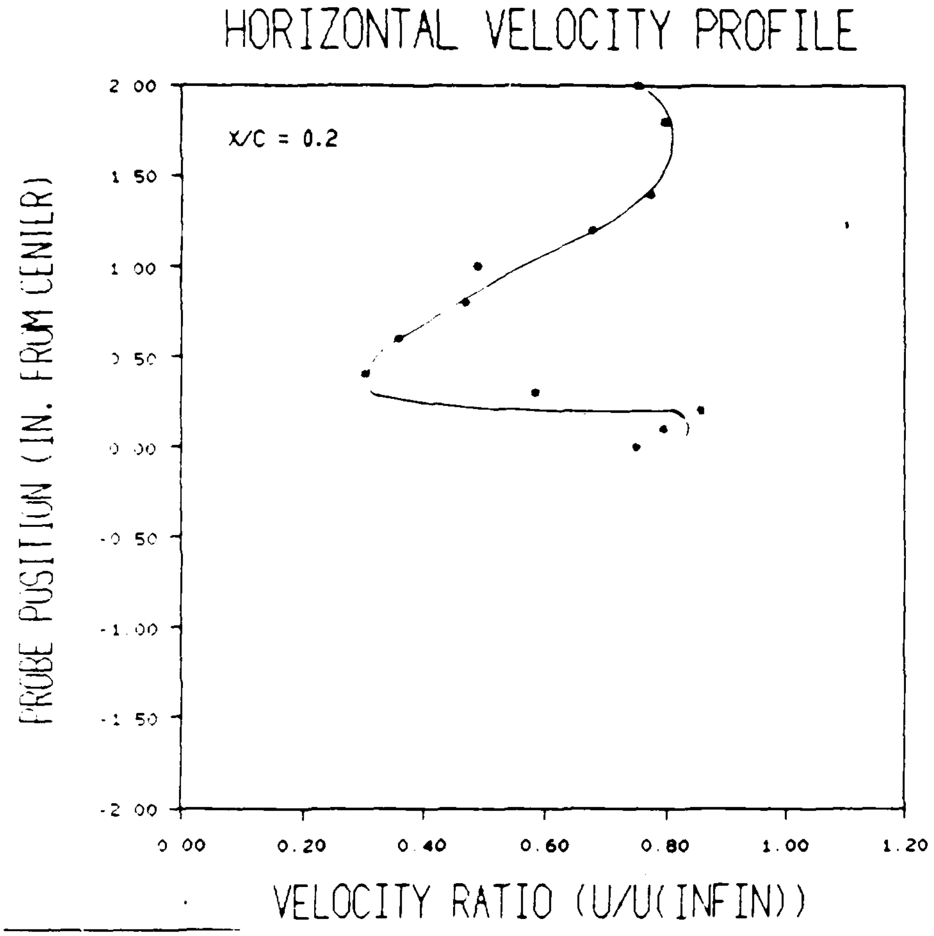


Figure 4.5 Steady wake profile.

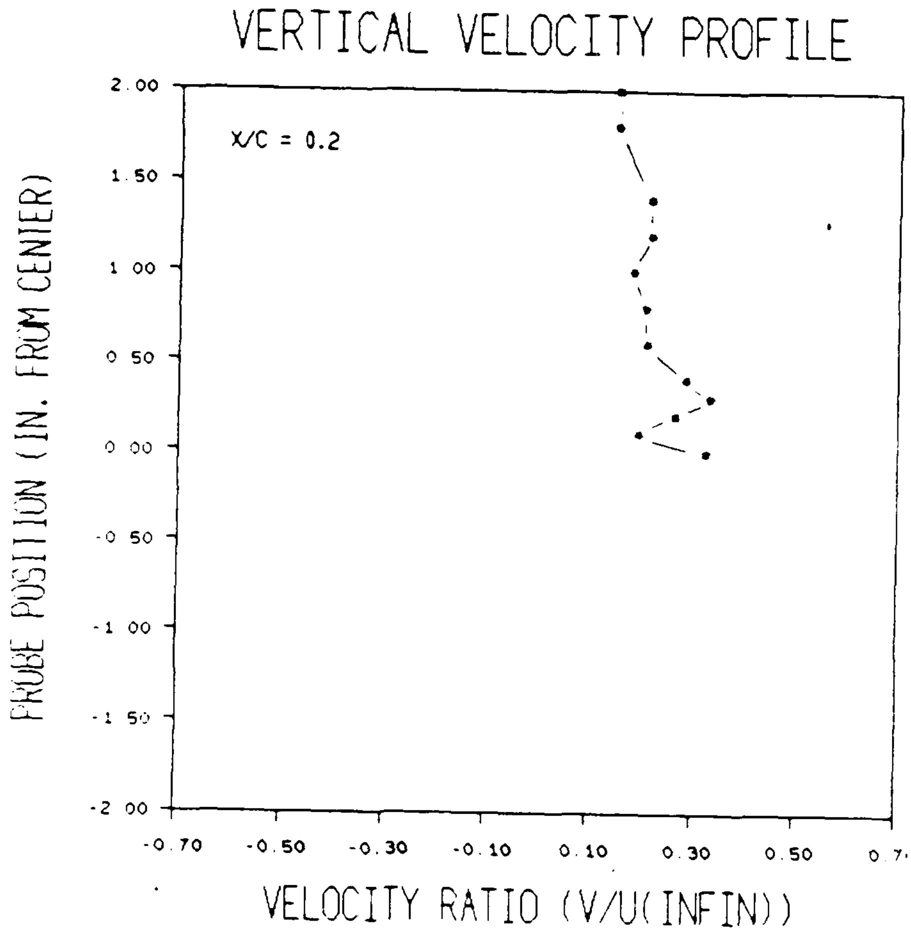


Figure 4.6 Steady wake profile.

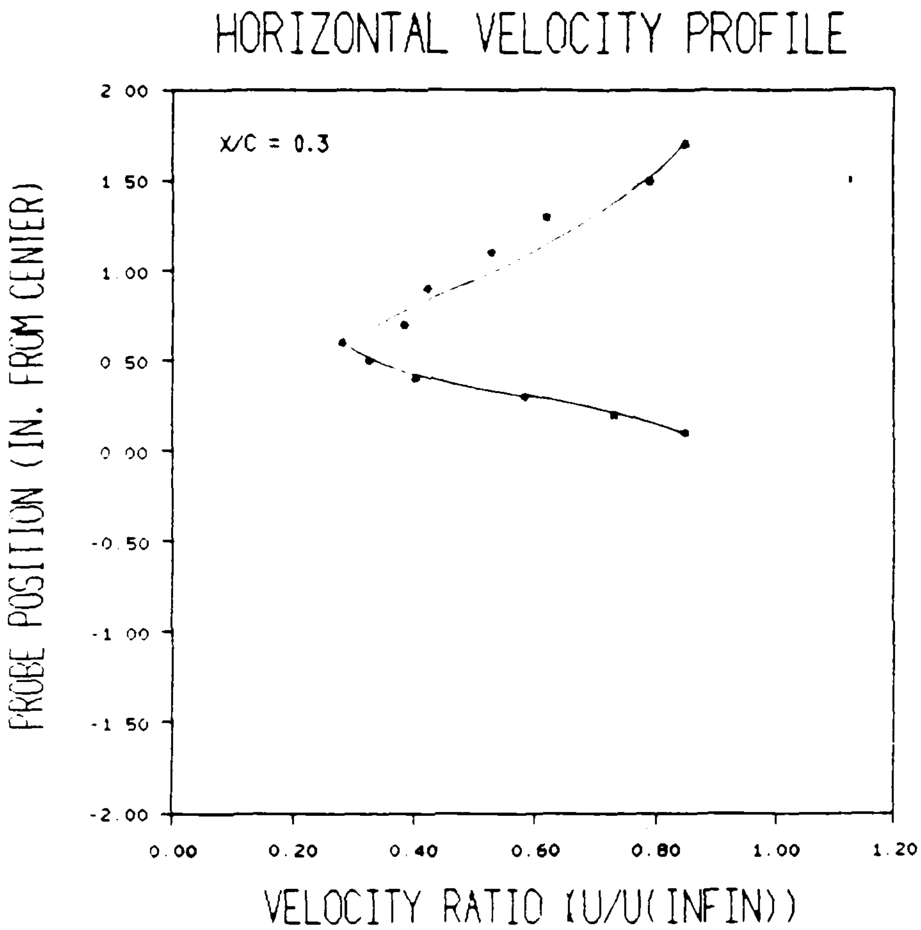


Figure 4.7 Steady wake profile.

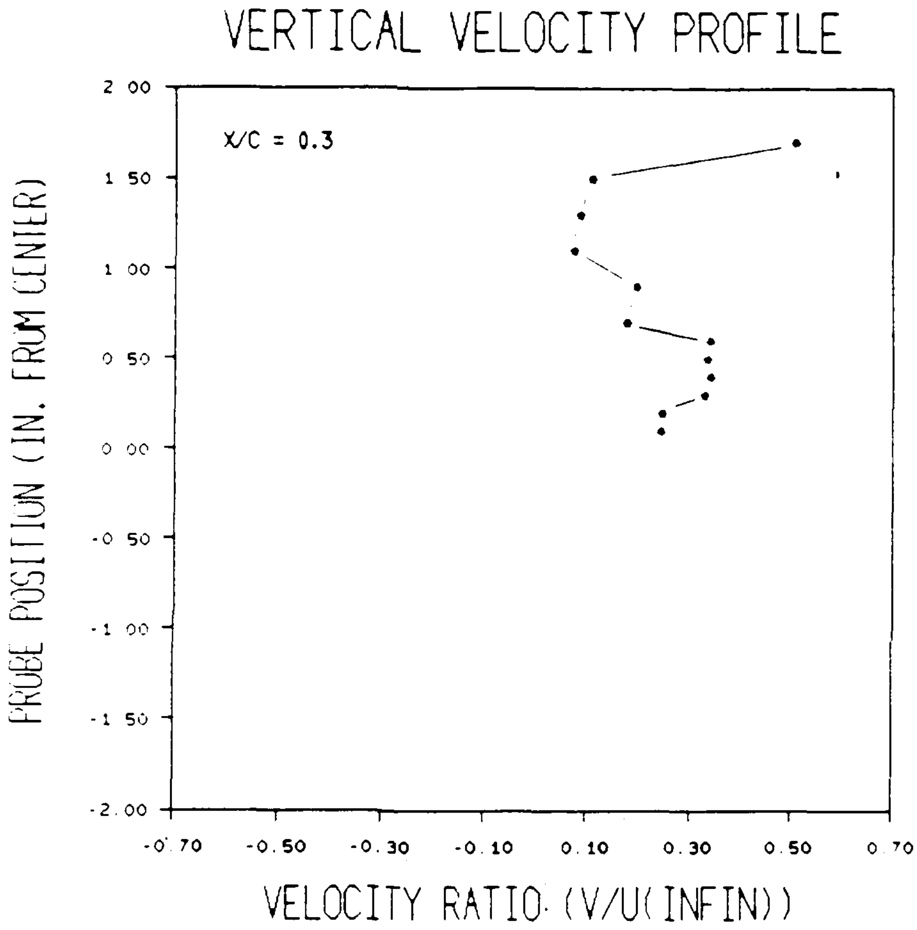


Figure 4.8 Steady wake profile.

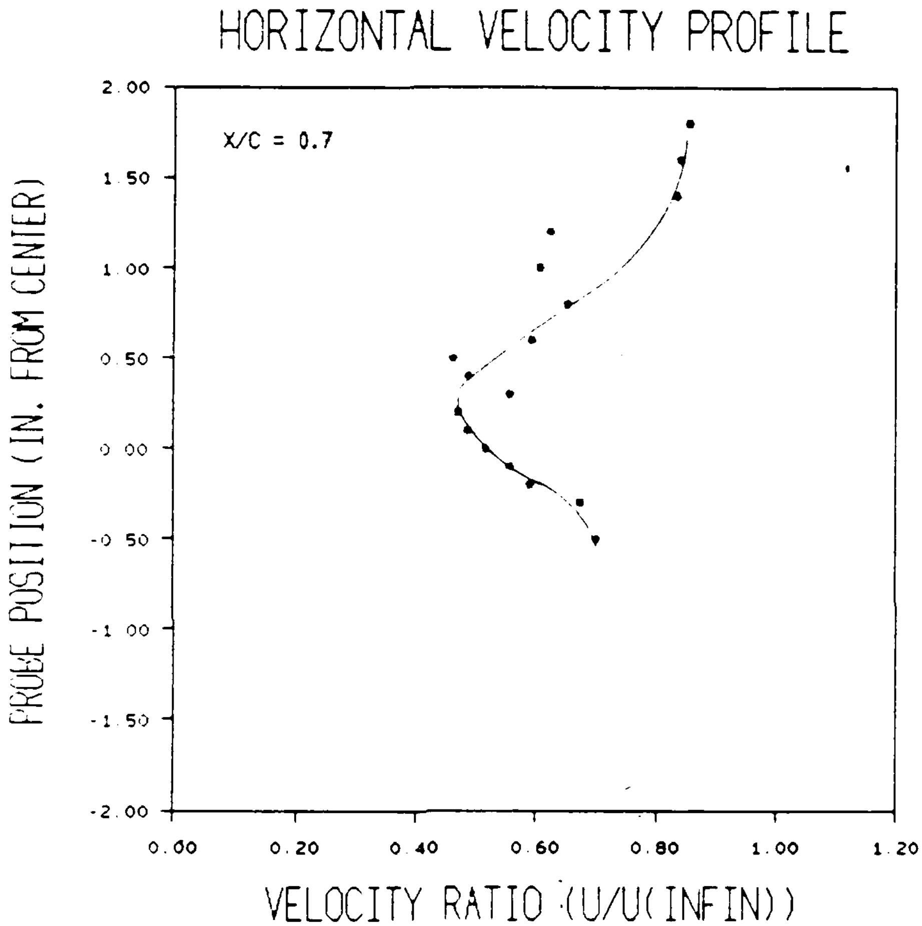


Figure 4.9 Steady wake profile.

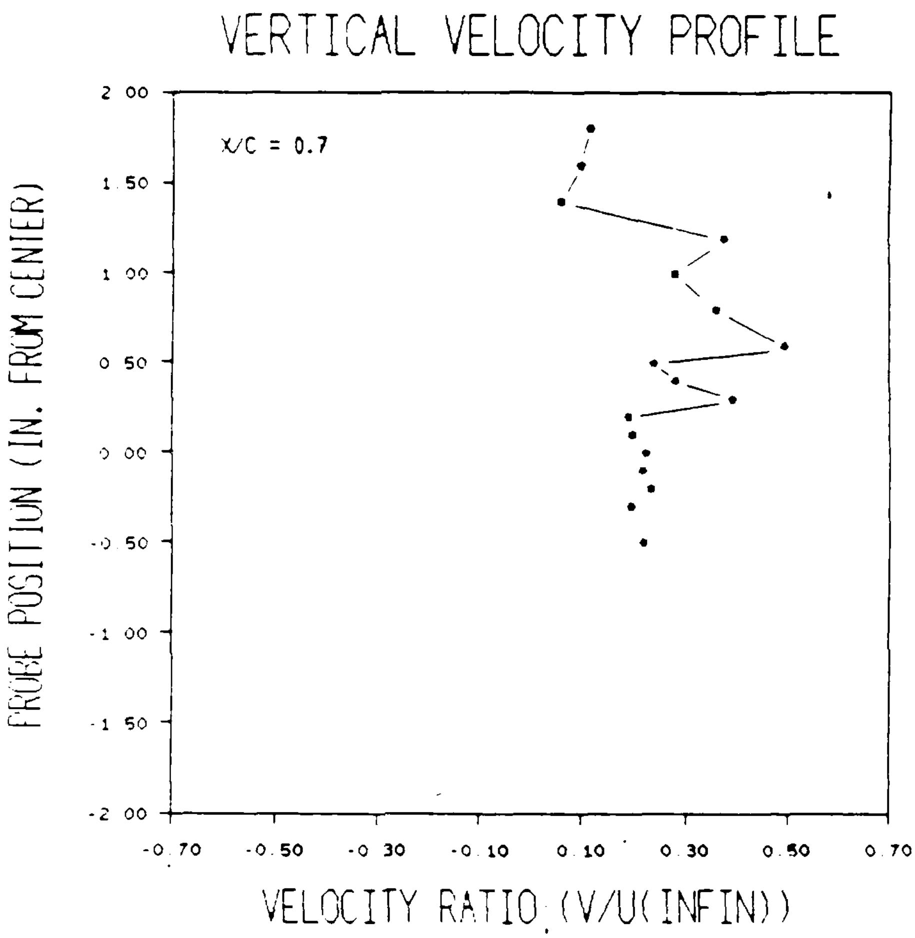


Figure 4.10 Steady wake profile.

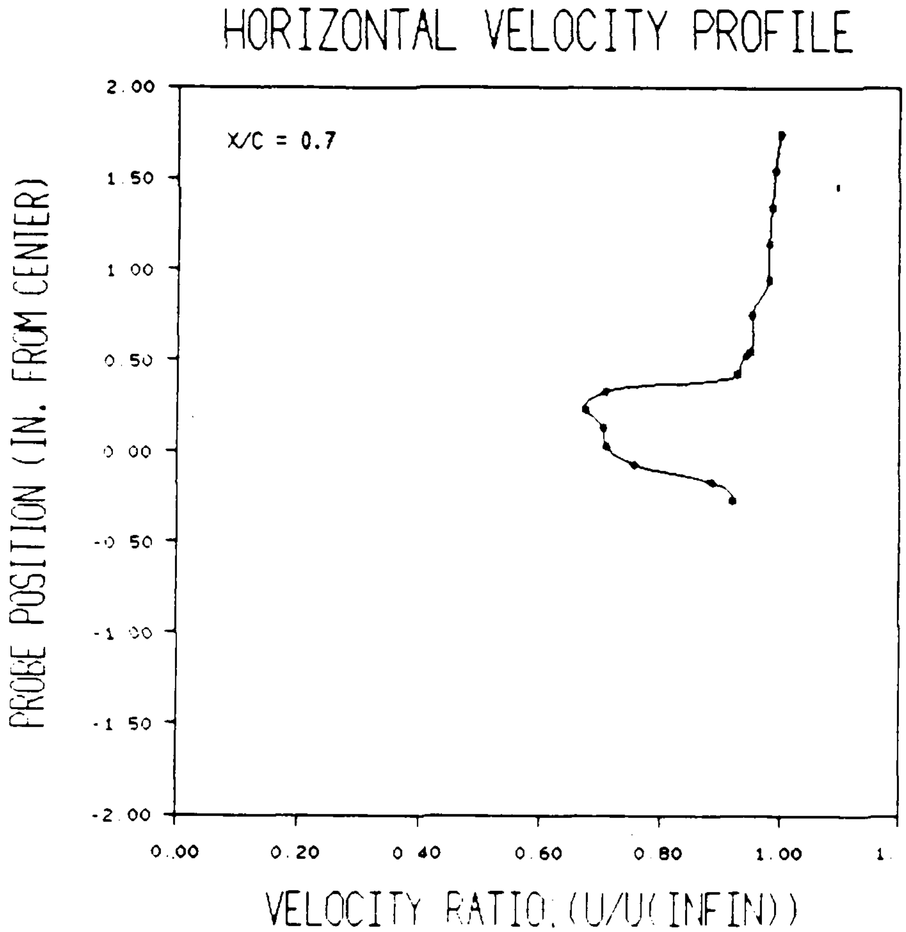


Figure 4.11 Unsteady wake profile.

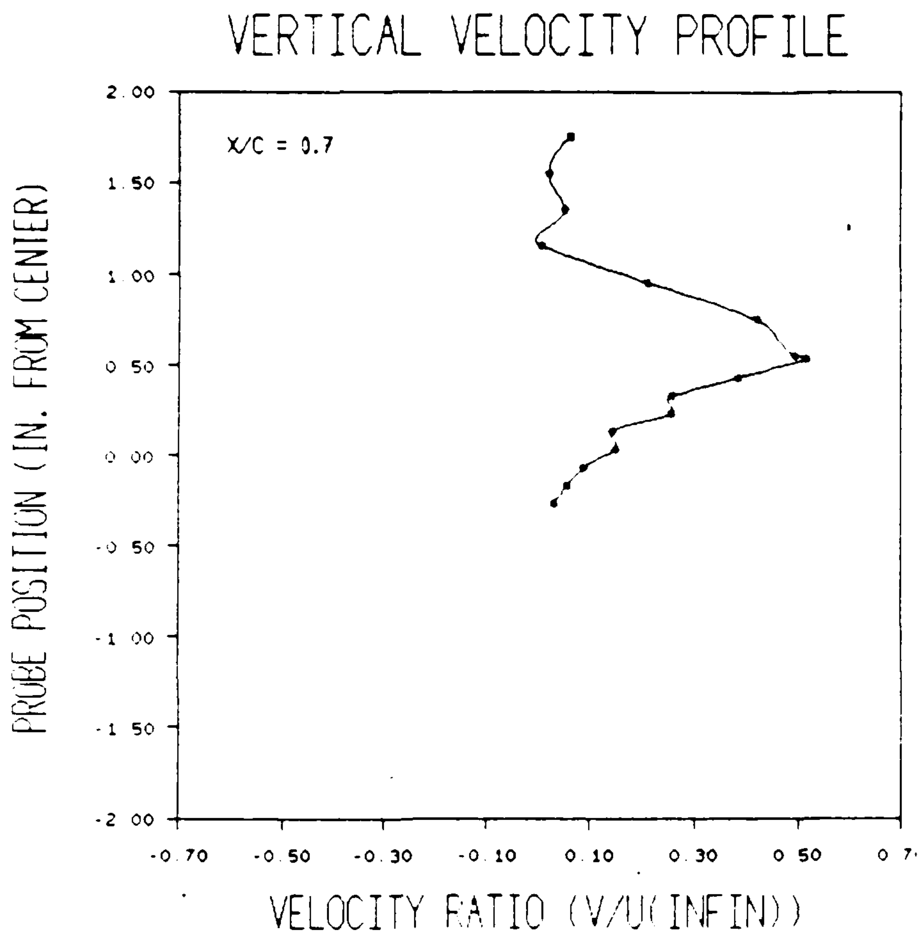


Figure 4.12 Unsteady wake profile.

V. CONCLUSIONS AND RECOMMENDATIONS

The purpose of the results obtained during the experimental phase of this thesis was to validate the technique and software developed. This was accomplished when unsteady data, generated by an oscillating airfoil, was successfully acquired, analyzed, and reduced to wake profile plots.

Throughout the course of this investigation hardware problems have a posed a signigicant challenge to acquiring data within the existing time constraints. Consequently, the data obtained was accepted and included in this report without the opportunity to more carefully investigate regions of significant data scatter and improve, or validate, the data plots.

The results produced by these procedures confirm a sound technique, but also reveal several areas in need of additional emphasis. At present, an interpolation subroutine is required to produce data at equally spaced intervals before generation of the curve fitting coefficients by 'FOURIER'. Combined with the inability to utilize a large number of data points (< 300), this limitation will inherently introduce additional uncertainty into the results. A second software modification to be considered is the generation of the velocity data at various phase angles simultaneously. Currently, the data must be processed for each separate phase angle

desired. This would eliminate the need for storage of large amounts of raw data and significantly reduce the processing time. By increasing the range of vertical movement of the traverse, the mechanical restriction that prevented the acquisition of data in the lower regions of the wake can be eliminated.

Continuation of this study will be accomplished with a newly developed test section currently being installed. This test section will enable schlieren flow visualization and LDV data acquisition at locations over the airfoil which were previously blocked by the mechanical linkage used for supporting and oscillating the airfoil. This, combined with the modifications suggested above, will enable an in-depth analysis of wake and boundary layer characteristics associated with the dynamic stall phenomenon.

APPENDIX

COMPUTER PROGRAMS

```
      PROGRAM LDV_ACQ
***************************************************************************
*                                                                         *
*  Adapted from LABSTAR Example3_1 by S. D. Hedrick in May, 1987 for reading  *
*  A-D data.  Modified in August, 1987 to be used as the central program in   *
*  the acquisition of unsteady velocity components with an LDV.               *
*                                                                         *
***************************************************************************

      INCLUDE 'sys$library:LIOSET.FOR'
C declare local variables
      INTEGER sys_stat          !status ret by LIO calls
      INTEGER dev_id, clk_id    !LIO device ID
      INTEGER data_length       !number of bytes of data read
      INTEGER i, npt, dnpt
      INTEGER chn, nsteps, ier
      REAL    rate, volt(0:23), press(0:23), corfac
      REAL    sum, mean, calp, conv, amb, temp, mach, velo
      CHARACTER*1 ans
      CHARACTER*4 clock
      LOGICAL home

C declare data buffers
      INTEGER*2 raw_data(501)
      REAL   VOLTAGES(501)

      OPEN(UNIT=13,FILE='scani.dat',STATUS='NEW')
C
      write(*,*) 'ENTER A-D CHANNEL TO BE USED (0,1,8,9)'
      read(*,*) chn
  10  write(*,*) 'ENTER CLOCK RATE (.01 - 4500 Hz)--ALLOW FOR A',
     1         ' MAXIMUM OF 2 SECONDS TO ACQUIRE DESIRED SAMPLES'
      read(*,*) rate
      if (rate .lt. .01 .or. rate .gt. 4500) then
        write(*,*) 'RATE IS NOT WITHIN SPECIFIED LIMITS'
        go to 10
      end if
  20  write(*,*)'ENTER NUMBER OF DATA POINTS TO BE COLLECTED',
     1          ' (500 max)'

      read(*,*) npt
      if (npt .lt. 1 .or. npt .gt. 500) then
        write(*,*) 'NUMBER OR POINTS IS NOT WITHIN LIMITS'
        go to 20
      end if
      npt = npt + 1
      dnpt = npt * 2
      write(*,*) 'ENTER CLOCK DEVICE (KZAO,KZBO)'
      read(*,'(A4)') clock
C
C Attach the ADV11-D and set up for queued I/O
C
      sys_stat=LIO$ATTACH(dev_id,'AZA0', LIO$K_QIO)
          IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
C
C Set up the ADV for synchronous transfer on selected channel
C      Gain of 1
C
      sys_stat = LIO$SET_I (dev_id, LIO$K_SYNCH, 0)
          IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
```

```
      sys_stat = LIO$SET_I (dev_id, LIO$K_AD_CHAN, 1, chn)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))

      sys_stat = LIO$SET_I (dev_id, LIO$K_AD_GAIN, 1, 1)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
C
C Get a raw_data buffer
C      length of buffer and returned data length is in bytes
C Attach clock and set rate
C
      home = .true.
      nsteps = 0
      do 40 j=0,7
      call dalljllab(home,nsteps,ier)
      sys_stat= LIO$ATTACH(clk_id,clock,LIO$K_QIO)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
      sys_stat=LIO$SET_R(clk_id,LIO$K_CLK_RATE, 1,rate)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
      sys_stat=LIO$SET_I(clk_id,LIO$K_TRIG, 1, LIO$K_IMMEDIATE)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
      sys_stat = LIO$READ(dev_id, raw_data,dnpt,data_length, )
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
      sys_stat = LIO$SET_I (dev_id, LIO$K_TRIG, 2, LIO$K_CLK_POINT,
     1             clk_id)
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
C
C Convert the raw data to voltages
C
      call LSP$FORMAT_TRANSLATE_ADC(raw_data,voltages,npt, , , )
C
C
C calculate the mean value of the voltages and detach the clock
C
      sum = 0.0
      do 30 i=2,npt
        sum = sum + voltages(i)
 30   continue
      mean = sum / float(npt-1)
      volt(j) = mean
      sys_stat = LIO$DETACH(clk_id, )
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
 40   continue
C
C detach from the A/D
C
      sys_stat = LIO$DETACH(dev_id, )
              IF(.NOT. sys_stat) CALL LIB$SIGNAL(%val(sys_stat))
C
C convert scanivalve voltages into units of pressure
C
      write(*,*) 'ENTER CALIBRATION PRESSURE (in. of Hg)'
      read(*,*) calp
      write(*,*) 'ENTER REQUIRED CORRECTION FACTOR FOR CALIBRATION',
     1           ' PRESSURE (in. of Hg)'
      read(*,*) corfac
      calp=calp+corfac
      write(*,*) 'ENTER ATMOSPHERIC PRESSURE (in. of Hg)'
      read(*,*) amb
      press(0) = calp
```

```
      press(1) = amb
      conv = (press(1)-press(0)) / (volt(1)-volt(0))
      do 50 j=2,7
        press(j) = conv * volt(j) + calp - volt(0) * conv
 50   continue
      WRITE(*,*) VOLT(0),VOLT(1),VOLT(2),VOLT(7)
      write(13,55)
 55   format(2X,'PORT',4X,'PRESSURE',//)
      do 60 i=0,7
        write(13,'(3X,I2,5X,F9.4)') i, press(i)
 60   continue
C
C compute test section velocity
C
      write(*,*)'ENTER AMBIENT TEMPERATURE (deg. F)'
      read(*,*) temp
      mach=sqrt(abs(5.*((press(2)/press(7))**(.4/1.4)-1.)))
      velo=mach*sqrt(1.4*53.3*32.174*(temp+459.67))
      write(*,*)'TEST SECTION MACH NUMBER AND VELOCITY--',mach,velo
C
      write(*,*)'CONTINUE WITH LDV DATA ACQUISITION? (Y/N, DEF=Y)'
      read(*,'(A1)') ans
      if (ans .eq. 'n' .or. ans .eq. 'N') go to 70
C
C call subroutine to move LDV traversing mechanism and acquire data
C
      call HWTRVT(velo)
C
 70   stop
      end
```

```
      SUBROUTINE HWWA(XPOS,ZPOS,VELO)
C
C ACRONYM:  HardWare checkout - test of DRV11-WA
C           -   -                         --
C PURPOSE:
C       For LVIS, HWDVYB drives the acquisition of the LDV data.
C       It is a test program for the YB driver.
C
C METHOD:
C
C  BEGIN
C  Install loadable drivers if necessary.
C  Read the various instrument types in succession, skipping those which
C     are not to be read.
C  Remove loadable drivers if necessary.
C  END
C
C PARAMETERS:
C    ARG        DIM       TYPE I/O/S DESCRIPTION
C
C COMMONS USED:
C
C FILES USED:  None
C
C ERROR HANDLING:
C    IER = 1 means no "fatal" error.
C
C NOTES:
C
C LOCAL VARIABLES:
C    VAR        DIM       TYPE    DESCRIPTION
C
C EXTERNAL REFERENCES:
C    NAME       DESCRIPTION AND SOURCE
C    DVWA       Calls DRV11-WA device driver to acquire A/D data.
C
C STANDARDS VIOLATIONS: None.
C
C ENVIRONMENT:  DEC VAX/VMS AND FORTRAN 77
C
C DEVELOPMENT HISTORY:
C         DATE   INITIALS    DESCRIPTION
C      07/26/83    TML      Adapted from DAPNT
C      12/12/85    CLH      Modified to print results in either
C                           octal or integer.
C      02/25/87    GBG      Modified for use with DRV11-WA and VMS
C      05/06/87    CLH      Modified to accept "new" DAMUX1.
C      06/08/87    RRR      Modified to subroutine + deleted CLH 5/87 Mod
C   JUL-AUG,1987   SDH      Modified to calculate velocities using variable
C                           mixing frequency as user input, interpret phase
C                           angle information generated by digital encoder,
C                           transfer test section velocity and probe posit-
C                           ion to data file, and generate histogram arrays
C                           and scaling parameters
C
C AUTHOR(S): Ted Lichtenstein, Informatics General Corp.
C
C-**********************************************************************
C
      IMPLICIT NONE
```

```
C
      PARAMETER          NRAWMX=80000
      PARAMETER          LUNTI=5
      PARAMETER          LUNOUT=6
C
      INTEGER*2          IRAWDT(NRAWMX), IFORM, MUXTIME (16384)
      INTEGER*2          MUXSTS (16384),RAWDA(20000),RAWDB(20000)
      INTEGER*2          RAWDC(20000)
        INTEGER          STATUS,IER,PHAN
        INTEGER          NCOLUM, IYESNO, NRAWIN,I,J
        INTEGER          LIB$SUBX, NCHANS, NSAMPLES, INDEX, JNDEX
        INTEGER          MRAWA,MRAWB,MRAWC,MRAWD
        INTEGER          IRAWDTA,IRAWDTB,IRAWDTC,IRAWDTD
        INTEGER          IRAWDTE,IRAWDTF
        REAL             FREQ(2),VEL(2),MIXA,MIXB,XPOS,ZPOS,VELO
        LOGICAL          DATSELCT (3)
      CHARACTER*10       INFILE
      CHARACTER*1        ANS
        REAL*4           YU(160),YV(120),YU1(160),YV1(120),YMAXU,YMAXV
        REAL*4           SAVE
        INTEGER          COUNT,XLOW,XHI
        REAL             X(4000),Y(4000)
C
        INCLUDE '($SYSSRVNAM)'
C
 100  WRITE(*,*)'ENTER DESIRED NAME OF DATA FILE--E.G., LDVXXX.DAT,',
     1           ' WHERE THE "XXX" MAY BE USED AS A SERIAL ID FOR THE',
     1           ' PROBE POSITION'
      READ(*,'(A10)') INFILE
      open (unit=11,file=INFILE,status='new')

      CONTINUE
C  *  Fill the raw data buffer with this bit pattern:
C  *  "1010101010101010"
C  *  This equals 125252(8), -21846(10).
C
      DO 200 I = 1, NRAWMX
         IRAWDT(I) = '125252'O
 200  CONTINUE

        DO I = 1, 16384

                MUXTIME (I) = '125252'O
                MUXSTS (I) = '125252'O

        END DO


C
 220            CONTINUE
C  *  Prompt for no. of words to acquire:
C
                WRITE(*,*)' '
                WRITE(*,*)'Starting LDV acquisition'
                WRITE(*,*)' '
                WRITE(*,*)'ENTER NUMBER OF CHANNELS TO BE USED',
     1                    ' (assumes non-oscillating airfoil if',
     1                    ' less than six)'
                READ(*,*) NCHANS
C
```

```
224             WRITE (*, 225)
225             FORMAT (' Enter number of samples:')
                READ (*, *) NSAMPLES

                IF (NSAMPLES .GT. 20000) THEN
                       WRITE (*, 226)
226             FORMAT (' The number of samples must be 20,000 or less.')
                       GO TO 224
                END IF


C
 230   CONTINUE
C  *   Prompt for output format of data:
C
       IFORM=2
       IF ( IFORM.LT.1 .OR. IFORM.GT.2 ) GO TO 230
C
C
C
C        Select the data to be returned.
C

                DATSELCT (1) = .false.
                DATSELCT (2) = .false.
                DATSELCT (3) = .TRUE.
C
C        Prompt for the mixing frequencies selected by user
C
       WRITE(*,*)'ENTER MIXING FREQUENCIES FOR EACH COMPONENT - (blue,
      1green) - in MHz'
       READ(*,*) MIXA,MIXB

C
C        Read and display the data one sample at a time.
C
       IF (NCHANS .LT. 6) THEN
         WRITE(*,*)'ENTER AIRFOIL PHASE ANGLE (tenths of degrees, I4)'
         READ(*,*) PHAN
         DO 9 I=1,NSAMPLES
           CALL DAMUX(NCHANS,1,DATSELCT,MUXTIME,MUXSTS,IRAWDT,IER)
           RAWDA(I)=IRAWDT(1)
           RAWDB(I)=IRAWDT(2)
           RAWDC(I)=PHAN
   9     CONTINUE
         GO TO 15
       END IF
                DO 10 I=1,NSAMPLES

                       CALL DAMUX (NCHANS, 1, DATSELCT, MUXTIME, MUXSTS,
      *                        IRAWDT, IER)
                       RAWDA(I)=IRAWDT(1)
                       RAWDB(I)=IRAWDT(2)
                       RAWDC(I)=IRAWDT(3)
  10            CONTINUE
C
                DO 19 I=1,NSAMPLES
                  MRAWA=0
```

```
                 MRAWB=0
                 MRAWC=0
                 MRAWD=0
                 MRAWA=IIBITS(RAWDC(I),12,2)
                 MRAWB=IIBITS(RAWDC(I),8,4)
                 MRAWC=IIBITS(RAWDC(I),4,4)
                 MRAWD=IIBITS(RAWDC(I),0,4)
                 RAWDC(I)=MRAWA*1000+MRAWB*100+MRAWC*10+MRAWD
 19            CONTINUE
 15            DO 20 I=1,NSAMPLES
                 RAWDA(I)=INOT(RAWDA(I))
                 RAWDB(I)=INOT(RAWDB(I))
                 IRAWDTA=0
                 IRAWDTB=0
                 IRAWDTC=0
                 IRAWDTD=0
                 IRAWDTE=0
                 IRAWDTF=0
                 IRAWDTA=IIBITS(RAWDA(I),10,4)
                 IRAWDTB=IIBITS(RAWDA(I),0,10)
                 IRAWDTC=2**IRAWDTA*IRAWDTB
                 IF (IRAWDTC .EQ. 0) THEN
                   WRITE(*,*)'IRAWDTC=0'
                   GO TO 20
                 END IF
                 FREQ(1)=((32.*10.**3.)/(FLOAT(IRAWDTC)))-40.+MIXA
                 IRAWDTD=IIBITS(RAWDB(I),10,4)
                 IRAWDTE=IIBITS(RAWDB(I),0,10)
                 IRAWDTF=2**IRAWDTD*IRAWDTE
                 IF (IRAWDTF .EQ. 0) THEN
                   WRITE(*,*)'IRAWDTF=0'
                   GO TO 20
                 END IF
                 FREQ(2)=((32.*10.**3.)/(FLOAT(IRAWDTF)))-40.+MIXB
                 vel(1)=freq(2)*0.5145/(2.*sin(atan(13./(2.*482.6))))
                 vel(2)=freq(1)*0.4880/(2.*sin(atan(13./(2.*482.6))))
                 vel(1)=vel(1)/.3048
                 vel(2)=vel(2)/.3048
C
                 WRITE(11,700) vel(2),vel(1),RAWDC(I)
 20            CONTINUE
      WRITE(11,900) XPOS,ZPOS
      WRITE(11,950) VELO,NSAMPLES
      WRITE(11,975) MIXA,MIXB
C
C
      WRITE(*,*)'DO YOU DESIRE A HISTOGRAM OF THE VELOCITIES OBTAINED?',
     1          ' (Y/N - DEF=N)'
      READ(*,'(A1)') ANS
      IF (ANS .NE. 'y' .AND. ANS .NE. 'Y') THEN
        GO TO 35
      END IF
      CLOSE(UNIT=11)
      OPEN(UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 40 I=1,NSAMPLES
        READ(11,800) X(I),Y(I)
 40   CONTINUE
      XLOW=-200
      XHI=-195
      DO 60 J=1,160
```

```
          COUNT=0
          DO 50 I=1,NSAMPLES
            IF (X(I) .GE. XLOW .AND. X(I) .LT. XHI) THEN
              COUNT=COUNT+1
            END IF
   50     CONTINUE
          YU(J)=(FLOAT(COUNT)/NSAMPLES)*100.
          XLOW=XLOW+5
          XHI=XHI+5
   60   CONTINUE
C
C  Calculating a scaling parameter for the horizontal velocity histogram
C
        DO 62 I=1,160
          YU1(I)=YU(I)
   62   CONTINUE
        DO 63 I=1,160
          DO 65 J=1,159
            IF (YU1(J+1) .LT. YU1(J)) THEN
              SAVE=YU1(J)
              YU1(J)=YU1(J+1)
              YU1(J+1)=SAVE
            END IF
   65     CONTINUE
   63   CONTINUE
        YMAXU=YU1(160)+10.0
C
        XLOW=-300
        XHI=-295
        DO 80 J=1,120
          COUNT=0
          DO 70 I=1,NSAMPLES
            IF (Y(I) .GE. XLOW .AND. Y(I) .LT. XHI) THEN
              COUNT=COUNT+1
            END IF
   70     CONTINUE
          YV(J)=(FLOAT(COUNT)/NSAMPLES)*100.
          XLOW=XLOW+5
          XHI=XHI+5
   80   CONTINUE
C
C  Calculating a scaling parameter for the vertical velocity histogram
C
        DO 82 I=1,120
          YV1(I)=YV(I)
   82   CONTINUE
        DO 83 I=1,120
          DO 85 J=1,119
            IF (YV1(J+1) .LT. YV1(J)) THEN
              SAVE=YV1(J)
              YV1(J)=YV1(J+1)
              YV1(J+1)=SAVE
            END IF
   85     CONTINUE
   83   CONTINUE
        YMAXV=YV1(120)+10.0
C
        CALL HIST(YU,YV,YMAXU,YMAXV)
C
   35   CLOSE(UNIT=11)
```

```
C
      WRITE(*,*)'DO YOU DESIRE A PLOT OF THE COMPUTED VELOCITY',
     1              ' VALUES? (Y/N - DEF=Y)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') THEN
        GO TO 30
      END IF
      CALL PT_PLOT(NSAMPLES,INFILE)
      WRITE(*,*)'DO YOU DESIRE TO VARY THE PLOTTED VELOCITY RANGE?',
     1              ' (Y/N--DEF=Y)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') GO TO 30
      CALL PPLOT(NSAMPLES,INFILE)
C  *  Again?
   30 WRITE(*,*)' '
      WRITE (LUNTI,*) 'Again?  (0=NO, 1=YES) '
      READ  (LUNTI,*) IYESNO
      IF ( IYESNO.EQ.1 ) GO TO 100
C
      RETURN

C  *  Formats

  700 FORMAT(5X,F9.4,1X,F9.4,8X,I4)
  800 FORMAT(5X,F9.4,1X,F9.4)
  900 FORMAT(//,2X,'XPOS=',F6.2,2X,'ZPOS=',F6.2)
  950 FORMAT(2X,'TEST SECTION VELOCITY=',F5.1,2X,I5,' SAMPLES')
  975 FORMAT(2X,'BLUE AND GREEN MIXING FREQUENCIES--',2F5.1,'(MHz)')
      END
```

```
      SUBROUTINE PT_PLOT(NSAMPLES,INFILE)
*************************************************************************
*                                                                       *
*  Written by S. D. Hedrick in July, 1987 to plot computed velocity values *
*  corresponding to airfoil phase angles.  This subroutine is intended to *
*  provide the user with graphical information on acquired LDV velocities *
*  and assist in making a determination regarding the adequacy of the data *
*  obtained at the current probe position.                              *
*                                                                       *
*************************************************************************
      IMPLICIT NONE
      INTEGER  NSAMPLES,I
      REAL*4   XVAL(4000),YVALU(4000),YVALV(4000),XCONTR(4),YCONTR(4)
      CHARACTER*10 INFILE
      CHARACTER*11 XLABEL
      CHARACTER*15 YLABEL
      CHARACTER*19 TITLE
      CHARACTER*1  DUMMY
      OPEN (UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 10 I=1,NSAMPLES
        READ(11,100) YVALU(I),YVALV(I),XVAL(I)
        XVAL(I)=XVAL(I)/10.0
 10   CONTINUE
100   FORMAT(5X,F9.4,1X,F9.4,8X,F4.0)
C
C  Set up the coordinate axis
C
      XLABEL='PHASE ANGLE'
      YLABEL='VELOCITY (ft/s)'
      TITLE='COMPUTED VELOCITIES'
      XCONTR(1)=6.0
      XCONTR(2)=0.0
      XCONTR(3)=360.0
      XCONTR(4)=45.0
      YCONTR(1)=6.0
      YCONTR(2)=-200.0
      YCONTR(3)=600.0
      YCONTR(4)=100.0
      CALL LGP$PLOT(1,'EXSY',XVAL,YVALU,0,XLABEL,YLABEL, , , ,XCONTR,
     1              YCONTR, ,TITLE)
C
C  Plot the individual data points on the axis
C
      CALL LGP$POINT(1,XVAL,YVALU,.03,3,NSAMPLES, , , )
      CALL LGP$POINT(1,XVAL,YVALV,.03,4,NSAMPLES, , , )
      CALL LGP$PUT_TEXT(1,4.5,5.75,'*  U-COMPONENT', , )
      CALL LGP$PUT_TEXT(1,4.5,5.5,'0  V-COMPONENT', , )
C
      WRITE(*,*) 'Press <CR> to terminate plot'
      READ(*,200) DUMMY
  200 FORMAT(A1)
      CALL LGP$TERMINATE_PLOT(1,1)
      CLOSE(UNIT=11)
      RETURN
      END
```

```
      SUBROUTINE PPLOT(NSAMPLES,INFILE)
      IMPLICIT NONE
      INTEGER  NSAMPLES,I
      REAL     VMIN,VMAX,INTV
      REAL*4   XVAL(4000),YVALU(4000),YVALV(4000),XCONTR(4),YCONTR(4)
      CHARACTER*10 INFILE
      CHARACTER*11 XLABEL
      CHARACTER*15 YLABEL
      CHARACTER*19 TITLE
      CHARACTER*1  DUMMY
      OPEN (UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 10 I=1,NSAMPLES
        READ(11,100) YVALU(I),YVALV(I),XVAL(I)
        XVAL(I)=XVAL(I)/10.0
   10 CONTINUE
  100 FORMAT(5X,F9.4,1X,F9.4,8X,F4.0)
C
C  Set up the coordinate axis
C
      WRITE(*,*)'ENTER VELOCITY RANGE DESIRED (min,max)'
      READ(*,*) VMIN,VMAX
      INTV=(VMAX-VMIN)/5.
      XLABEL='PHASE ANGLE'
      YLABEL='VELOCITY (ft/s)'
      TITLE='COMPUTED VELOCITIES'
      XCONTR(1)=6.0
      XCONTR(2)=0.0
      XCONTR(3)=360.0
      XCONTR(4)=45.0
      YCONTR(1)=6.0
      YCONTR(2)=VMIN
      YCONTR(3)=VMAX
      YCONTR(4)=INTV
      CALL LGP$PLOT(1,'EXSY',XVAL,YVALU,0,XLABEL,YLABEL, , , ,XCONTR,
     1              YCONTR, ,TITLE)
C
C  Plot the individual data points on the axis
C
      CALL LGP$POINT(1,XVAL,YVALU,.03,3,NSAMPLES, , , )
      CALL LGP$POINT(1,XVAL,YVALV,.03,4,NSAMPLES, , , )
      CALL LGP$PUT_TEXT(1,.05,.3,'ASTERISK=U COMPONENT', , )
      CALL LGP$PUT_TEXT(1,.05,.1,'CIRCLE=V COMPONENT', , )
C
      WRITE(*,*) 'Press <CR> to terminate plot'
      READ(*,200) DUMMY
  200 FORMAT(A1)
      CALL LGP$TERMINATE_PLOT(1,1)
      CLOSE(UNIT=11)
      RETURN
      END
```

```
      SUBROUTINE HIST(YU,YV,YMAXU,YMAXV)
*************************************************************************
*                                                                       *
*  Written by S. D. Hedrick in August, 1987 to plot a histogram of the per-  *
*  centage of data points which fall within specified velocity limits. The   *
*  subroutine produces histograms for each of two velocity components.  Scal- *
*  ing parameters for the vertical axis and data arrays are computed within  *
*  a preceding subroutine (HWWA).                                         *
*                                                                       *
*************************************************************************
      IMPLICIT NONE
      REAL*4 YU(160),YV(120),XCONTRU(4),XCONTRV(4),YCONTRU(4)
      REAL*4 YMAXU,YMAXV,YCONTRV(4)
      REAL XLOW,XHI,XLOWU(160),XHIU(160),XLOWV(120),XHIV(120)
      INTEGER I
      CHARACTER*29 UTITLE
      CHARACTER*27 VTITLE
      CHARACTER*10 XLABEL
      CHARACTER*18 YLABEL
      CHARACTER*1 DUMMY
C
C  Setting up the coordinate axis
C
      XLABEL='VELOCITIES'
      YLABEL='PERCENTAGE OF DATA'
      UTITLE='HORIZONTAL VELOCITY HISTOGRAM'
      VTITLE='VERTICAL VELOCITY HISTOGRAM'
      XCONTRU(1)=6.0
      XCONTRU(2)=-200.0
      XCONTRU(3)=600.0
      XCONTRU(4)=100.0
      XCONTRV(1)=6.0
      XCONTRV(2)=-300.0
      XCONTRV(3)=300.0
      XCONTRV(4)=100.0
      YCONTRU(1)=6.0
      YCONTRU(2)=0.0
      YCONTRU(3)=YMAXU
      YCONTRU(4)=10.0
      YCONTRV(1)=6.0
      YCONTRV(2)=0.0
      YCONTRV(3)=YMAXV
      YCONTRV(4)=10.0
      CALL LGP$PLOT(1,'IXSY', ,YU,0,XLABEL,YLABEL, , , ,XCONTRU,
     1              YCONTRU, ,UTITLE)
C
C  Setting up arrays for horizontal and vertical histogram bar parameters
C
      XLOW=-200.0
      XHI=-195.0
      DO 10 I=1,160
        XLOWU(I)=XLOW
        XHIU(I)=XHI
        XLOW=XLOW+5.0
        XHI=XHI+5.0
   10 CONTINUE
      XLOW=-300.0
      XHI=-295.0
      DO 20 I=1,120
        XLOWV(I)=XLOW
```

```
      XHIV(I)=XHI
      XLOW=XLOW+5.0
      XHI=XHI+5.0
   20 CONTINUE
C
C  Plotting the horizontal velocity histogram
C
      CALL LGP$HIST(1,XLOWU,XHIU,YU,160, , ,2)
C
      WRITE(*,*) 'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
C
C  Repeating the procedures to plot the vertical velocity histogram
C
      CALL LGP$PLOT(1,'IXSY', ,YV,0,XLABEL,YLABEL, , , ,XCONTRV,
     1              YCONTRV, ,VTITLE)
      CALL LGP$HIST(1,XLOWV,XHIV,YV,120, , ,2)
C
      WRITE(*,*) 'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
      RETURN
      END
```

```
      PROGRAM ORGIZE
******************************************************************************
*                                                                            *
*  Written by S. D. Hedrick in July, 1987 to sequentially organize data files *
*  to obtain a least value of phase angle (third column of data), and associ- *
*  ated velocities, at top of file.  This rearrangement of data was desired  *
*  to assist with subsequent manipulation and analysis.  In September, 1987  *
*  this routine was incorporated as the central program in the processing and *
*  analysis of unsteady LDV velocity data generated by an oscillating airfoil. *
*  The program calls assisting subroutines to disgard bad data points and pro- *
*  vide wake profile plots.                                                  *
*                                                                            *
******************************************************************************
      IMPLICIT NONE
      REAL X(4000),Y(4000),XPOS,ZPOS,VELO,UVL,VVL,UNDIM(30),VNDIM(30)
      REAL ZPO(30),SAV1,SAV2,MENU,MENV,PREJ,MEANU(18),MEANV(18)
      REAL MPHAVG(18)
      INTEGER Z(4000),NSAMPLES,PHASA,I,J,K,N,M,SAVE,REJ,STDY
      CHARACTER*10 INFILE
      CHARACTER*1 ANS
      STDY=0
      WRITE(*,*)'IS ANALYSIS FOR NON-OSCILLATING AIRFOIL? (Y/N)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') STDY=1
      WRITE(*,*)'ENTER NUMBER OF DATA ACQUISITION POINTS IN PROFILE',
     1          ' (30 maximum)'
      READ(*,*) N
      WRITE(*,*)'ENTER PHASE ANGLE DESIRED FOR ANALYSIS (degrees)'
      READ(*,*) PHASA
      DO 70 K=1,N
        WRITE(*,*)'ENTER NAME OF INPUT DATA FILE (LDVXXX.DAT)'
        READ(*,'(A10)') INFILE
        WRITE(*,*) 'ENTER NUMBER OF SAMPLES IN DATA FILE'
        READ(*,*) NSAMPLES
        OPEN(UNIT=11,FILE=INFILE,STATUS='OLD')
        DO 10 I=1,NSAMPLES
          READ(11,100) X(I),Y(I),Z(I)
10      CONTINUE
        READ(11,200) XPOS,ZPOS
        READ(11,300) VELO
        IF (STDY .EQ. 1) THEN
          CLOSE(UNIT=11)
          CALL STEADY(NSAMPLES,INFILE,MENU,MENV)
          UNDIM(K)=MENU/VELO
          VNDIM(K)=MENV/VELO
          GO TO 80
        END IF
        M=NSAMPLES-1
        DO 20 J=1,NSAMPLES
          DO 30 I=1,M
            IF (Z(I+1) .LT. Z(I)) THEN
              SAVE=Z(I)
              SAV1=X(I)
              SAV2=Y(I)
              Z(I)=Z(I+1)
              X(I)=X(I+1)
              Y(I)=Y(I+1)
              Z(I+1)=SAVE
              X(I+1)=SAV1
              Y(I+1)=SAV2
```

```
          END IF
30      CONTINUE
20    CONTINUE
      CLOSE(UNIT=11)
      OPEN(UNIT=15,FILE=INFILE,STATUS='OLD')
      DO 40 I=1,NSAMPLES
        WRITE(15,100) X(I),Y(I),Z(I)
40    CONTINUE
      CLOSE(UNIT=15)
      WRITE(*,*)'DO YOU DESIRE A VELOCITY/PHASE PLOT OF
     1           ' DATA?  (Y/N--DEF=Y)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') GO TO 50
      CALL PT_PLOT(NSAMPLES,INFILE)
50    CALL BDPTS(NSAMPLES,INFILE,MENU,MENV,REJ,PREJ,MEA
     1           MPHAVG)
      CALL AVG(NSAMPLES,INFILE)
      WRITE(*,*)'DO YOU DESIRE INTERPOLATION OF THE DAT
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') GO TO 55
      CALL INTERP(NSAMPLES,INFILE)
55    CALL CVFIT(NSAMPLES,INFILE,PHASA,UVL,VVL,MENU,MEN
     1           MEANU,MEANV,MPHAVG)
      UNDIM(K)=UVL/VELO
      VNDIM(K)=VVL/VELO
80    ZPO(K)=ZPOS
70  CONTINUE
    CALL PROPLOT(UNDIM,VNDIM,ZPO,XPOS,N)
C
C  Format statements
C
100 FORMAT(5X,F9.4,1X,F9.4,8X,I4)
200 FORMAT(//,7X,F6.2,7X,F6.2)
300 FORMAT(24X,F5.1)
    STOP
    END
```

```
      SUBROUTINE BDPTS(NSAMPLES,INFILE,MENU,MENV,REJ,PREJ,
     1                 MEANU,MEANV,MPHAVG)
      REAL X(4000),Y(4000),MEANU(18),MEANV(18),STDVU(18)
      REAL STDVV(18),MPHAVG(18),MINU,MINV,MAXU,MAXV,MENU,MENV
      INTEGER Z(4000),PHASEA,PHASEB,SUM,COUNT,NSAMPLES,NSAMP,REJ
      REAL*4 XCONTR(4),YCONTR(4),PREJ
      CHARACTER*1 ANS,DUMMY,DUMMI
      CHARACTER*10 INFILE
      CHARACTER*11 XLABEL
      CHARACTER*21 YLABEU
      CHARACTER*19 YLABEV
      CHARACTER*18 TITLE
      OPEN(UNIT=15,FILE=INFILE,STATUS='OLD')
      NSAMP=NSAMPLES
      DO 10 I=1,NSAMPLES
        READ(15,100) X(I),Y(I),Z(I)
   10 CONTINUE
  100 FORMAT(5X,F9.4,1X,F9.4,8X,I4)
      PHASEA=0
      PHASEB=200
      DO 20 I=1,18
        SUM=0
        SUMU=0.
        SUMV=0.
        PHAVG=0.
        DO 30 J=1,NSAMPLES
          IF (Z(J) .LE. PHASEB .AND. Z(J) .GE. PHASEA) THEN
            SUMU=SUMU+X(J)
            SUMV=SUMV+Y(J)
            PHAVG=PHAVG+(FLOAT(Z(J))/10.)
            SUM=SUM+1
          END IF
   30   CONTINUE
        IF (SUM .EQ. 0) THEN
          MEANU(I)=0.
          MEANV(I)=0.
          MPHAVG(I)=(FLOAT(PHASEA)+100.)/10.
          VARU=0.
          VARV=0.
          GO TO 40
        END IF
        MEANU(I)=SUMU/FLOAT(SUM)
        MEANV(I)=SUMV/FLOAT(SUM)
        MPHAVG(I)=PHAVG/FLOAT(SUM)
        VARU=0.
        VARV=0.
        DO 40 J=1,NSAMPLES
          IF (Z(J) .LE. PHASEB .AND. Z(J) .GE. PHASEA) THEN
            A1=X(J)-MEANU(I)
            A2=Y(J)-MEANV(I)
            B1=A1**2/FLOAT(SUM)
            B2=A2**2/FLOAT(SUM)
            VARU=VARU+B1
            VARV=VARV+B2
          END IF
   40   CONTINUE
        STDVU(I)=SQRT(VARU)
        STDVV(I)=SQRT(VARV)
        PHASEA=PHASEA+200
        PHASEB=PHASEB+200
```

```
   20  CONTINUE
       WRITE(*,*)'DO YOU DESIRE A STANDARD DEVIATION PLOT? (Y/N - DEF=Y)'
       READ(*,200) ANS
  200  FORMAT(A1)
       IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') THEN
         GO TO 50
       END IF
C
C  Setting up the coordinate axis for the plot
C
       XLABEL='PHASE ANGLE'
       YLABEU='HORIZONTAL VELOCITIES'
       YLABEV='VERTICAL VELOCITIES'
       TITLE='STANDARD DEVIATION'
       XCONTR(1)=6.0
       XCONTR(2)=0.0
       XCONTR(3)=360.0
       XCONTR(4)=45.0
       YCONTR(1)=6.0
       YCONTR(2)=-200.0
       YCONTR(3)=600.0
       YCONTR(4)=100.0
       CALL LGP$PLOT(1,'EXSY',MPHAVG,MEANU,0,XLABEL,YLABEU, , , ,XCONTR,
      1              YCONTR, ,TITLE)
C
C  Plotting the horizontal velocity data points on the axis
C
       CALL LGP$POINT(1,MPHAVG,MEANU,.03,4,18, , , )
C
C  Plotting the horizontal standard deviation
C
       CALL LGP$STNDEV(1,MPHAVG,MEANU,STDVU,18, )
C
   45  WRITE(*,*) 'PRESS <CR> TO TERMINATE PLOT'
       READ(*,200) DUMMY
       CALL LGP$TERMINATE_PLOT(1,1)
C
C  Plotting the vertical standard deviation
C
       CALL LGP$PLOT(1,'EXSY',MPHAVG,MEANV,0,XLABEL,YLABEV, , , ,XCONTR,
      1              YCONTR, ,TITLE)
       CALL LGP$POINT(1,MPHAVG,MEANV,.03,4,18, , , )
       CALL LGP$STNDEV(1,MPHAVG,MEANV,STDVV,18, )
   47  WRITE(*,*) 'PRESS <CR> TO TERMINATE PLOT'
       READ(*,200) DUMMI
       CALL LGP$TERMINATE_PLOT(1,1)
C
   50  COUNT=1
       PHASEA=0
       PHASEB=200
       DO 60 I=1,18
         MINU=MEANU(I)-2.0*STDVU(I)
         MINV=MEANV(I)-2.0*STDVV(I)
         MAXU=MEANU(I)+2.0*STDVU(I)
         MAXV=MEANV(I)+2.0*STDVV(I)
         DO 70 J=1,NSAMPLES
           IF (Z(J) .LE. PHASEB .AND. Z(J) .GE. PHASEA) THEN
             IF (X(J) .LT. MINU .OR. X(J) .GT. MAXU) THEN
               COUNT=COUNT+1
               IF (J .EQ. 1) GO TO 70
```

```
            DO 80 K=0,J-2
              X(J-K)=X(J-K-1)
              Y(J-K)=Y(J-K-1)
              Z(J-K)=Z(J-K-1)
80          CONTINUE
            GO TO 70
          END IF
          IF (Y(J) .LT. MINV .OR. Y(J) .GT. MAXV) THEN
            COUNT=COUNT+1
            IF (J .EQ. 1) GO TO 70
            DO 85 K=0,J-2
              X(J-K)=X(J-K-1)
              Y(J-K)=Y(J-K-1)
              Z(J-K)=Z(J-K-1)
85          CONTINUE
          END IF
        END IF
70    CONTINUE
      PHASEA=PHASEA+200
      PHASEB=PHASEB+200
60  CONTINUE
    CLOSE(UNIT=15)
    OPEN(UNIT=12,FILE=INFILE,STATUS='OLD')
    DO 90 I=COUNT,NSAMPLES
      WRITE(12,100) X(I),Y(I),Z(I)
90  CONTINUE
    NSAMPLES=NSAMPLES+1-COUNT
    CLOSE(UNIT=12)
    OPEN(UNIT=13,FILE=INFILE,STATUS='OLD')
    SUU=0.
    SUV=0.
    DO 95 I=1,NSAMPLES
      READ(13,100) X(I),Y(I),Z(I)
      SUU=SUU+X(I)
      SUV=SUV+Y(I)
95  CONTINUE
    MENU=SUU/FLOAT(NSAMPLES)
    MENV=SUV/FLOAT(NSAMPLES)
    CLOSE(UNIT=13)
    REJ=NSAMP-NSAMPLES
    PREJ=(1.-(FLOAT(NSAMPLES)/FLOAT(NSAMP)))*100.
    PHASEA=0
    PHASEB=200
    DO 96 I=1,18
      SUM=0
      SUMU=0.
      SUMV=0.
      PHAVG=0.
      DO 97 J=1,NSAMPLES
        IF (Z(J) .LE. PHASEB .AND. Z(J) .GE. PHASEA) THEN
          SUMU=SUMU+X(J)
          SUMV=SUMV+Y(J)
          PHAVG=PHAVG+(FLOAT(Z(J))/10.)
          SUM=SUM+1
        END IF
97    CONTINUE
      IF (SUM .EQ. 0) THEN
        MEANU(I)=0.
        MEANV(I)=0.
        MPHAVG(I)=(FLOAT(PHASEA)+100.)/10.
        GO TO 98
      END IF
      MEANU(I)=SUMU/FLOAT(SUM)
      MEANV(I)=SUMV/FLOAT(SUM)
      MPHAVG(I)=PHAVG/FLOAT(SUM)
98    PHASEA=PHASEA+200
      PHASEB=PHASEB+200
96  CONTINUE
    RETURN
    END
```

```
      SUBROUTINE AVG(NSAMPLES,INFILE)
      REAL  X(4000),Y(4000)
      INTEGER  Z(4000),COUNT,SUM,NSAMPLES
      CHARACTER*10 INFILE
      COUNT=1
      M=NSAMPLES-1
      OPEN(UNIT=12,FILE=INFILE,STATUS='OLD')
      DO 10 I=1,NSAMPLES
        READ(12,100) X(I),Y(I),Z(I)
 10   CONTINUE
100   FORMAT(5X,F9.4,1X,F9.4,8X,I4)
      DO 20 I=1,M
        IF (Z(I+1) .EQ. Z(I) .AND. Z(I+2) .NE. Z(I)) THEN
          COUNT=COUNT+1
          X(I+1)=(X(I+1)+X(I))/2.0
          Y(I+1)=(Y(I+1)+Y(I))/2.0
          DO 30 J=0,I-2
            X(I-J)=X(I-J-1)
            Y(I-J)=Y(I-J-1)
            Z(I-J)=Z(I-J-1)
 30       CONTINUE
        END IF
        IF (Z(I+2) .EQ. Z(I)) THEN
          SUM=1
          DO 25 J=2,NSAMPLES-I
            IF (Z(I+J) .EQ. Z(I)) THEN
              SUM=SUM+1
            END IF
 25       CONTINUE
          DO 27 J=1,SUM
            X(I+SUM)=X(I+SUM)+X(I+SUM-J)
            Y(I+SUM)=Y(I+SUM)+Y(I+SUM-J)
 27       CONTINUE
          X(I+SUM)=X(I+SUM)/(SUM+1)
          Y(I+SUM)=Y(I+SUM)/(SUM+1)
          DO 28 J=1,I-1
            X(I+SUM-J)=X(I-J)
            Y(I+SUM-J)=Y(I-J)
            Z(I+SUM-J)=Z(I-J)
 28       CONTINUE
          COUNT=COUNT+SUM
        END IF
 20   CONTINUE
      CLOSE(UNIT=12)
      OPEN(UNIT=14,FILE=INFILE,STATUS='OLD')
      DO 40 I=COUNT,NSAMPLES
        WRITE(14,100) X(I),Y(I),Z(I)
 40   CONTINUE
      NSAMPLES=NSAMPLES+1-COUNT
      CLOSE(UNIT=14)
      RETURN
      END
```

```
      SUBROUTINE INTERP(NSAMPLES,INFILE)
      IMPLICIT NONE
      CHARACTER*10  INFILE
      REAL  X(3600),Y(3600)
      INTEGER  Z(3600),NSAMPLES,I,J,COUNT,NSA,FAC
      CHARACTER*1  ANS
      OPEN(UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 10 I=1,NSAMPLES
        READ(11,100) X(I),Y(I),Z(I)
 10   CONTINUE
100   FORMAT(5X,F9.4,1X,F9.4,8X,I4)
      COUNT=0
      NSA=NSAMPLES
      DO 20 I=1,3600
        IF (COUNT .GE. NSA) THEN
          Z(I)=I-1
          X(I)=((X(1)-X(I-1))/(Z(1)-Z(I-1)))*(Z(I)-Z(I-1))+X(I-1)
          Y(I)=((Y(1)-Y(I-1))/(Z(1)-Z(I-1)))*(Z(I)-Z(I-1))+Y(I-1)
          NSAMPLES=NSAMPLES+1
          GO TO 20
        END IF
        IF (Z(I) .EQ. I-1) COUNT=COUNT+1
        IF (Z(I) .NE. I-1) THEN
          NSAMPLES=NSAMPLES+1
          DO 30 J=0,NSAMPLES-I-1
            X(NSAMPLES-J)=X(NSAMPLES-J-1)
            Y(NSAMPLES-J)=Y(NSAMPLES-J-1)
            Z(NSAMPLES-J)=Z(NSAMPLES-J-1)
 30       CONTINUE
          Z(I)=I-1
          IF (I .EQ. 1) THEN
            X(I)=((X(I+1)-X(NSAMPLES))/(Z(I+1)-Z(NSAMPLES)))*(Z(I)-
     1           Z(NSAMPLES))+X(NSAMPLES)
            Y(I)=((Y(I+1)-Y(NSAMPLES))/(Z(I+1)-Z(NSAMPLES)))*(Z(I)-
     1           Z(NSAMPLES))+Y(NSAMPLES)
            GO TO 20
          END IF
          X(I)=((X(I+1)-X(I-1))/(Z(I+1)-Z(I-1)))*(Z(I)-Z(I-1))+X(I-1)
          Y(I)=((Y(I+1)-Y(I-1))/(Z(I+1)-Z(I-1)))*(Z(I)-Z(I-1))+Y(I-1)
        END IF
 20   CONTINUE
      CLOSE(UNIT=11)
C      WRITE(*,*)'DO YOU DESIRE TO REDUCE THE NUMBER OF INTERPOLATED',
C     1      ' DATA POINTS BY AN INTEGER FACTOR OF 3600? (Y/N)'
C      READ(*,'(A1)') ANS
      OPEN(UNIT=12,FILE=INFILE,STATUS='OLD')
C      IF (ANS .EQ. 'Y' .OR. ANS .EQ. 'y') THEN
C        WRITE(*,*)'ENTER INTEGER FACTOR'
C        READ(*,*) FAC
      FAC=20
        DO 34 I=FAC,NSAMPLES,FAC
          WRITE(12,100) X(I),Y(I),Z(I)
 34     CONTINUE
        NSAMPLES=3600/FAC
C        GO TO 40
C      END IF
C      DO 40 I=1,NSAMPLES
C        WRITE(12,100) X(I),Y(I),Z(I)
C 40   CONTINUE
      CLOSE(UNIT=12)
      RETURN
      END
```

```
      SUBROUTINE FOURIER(F,M,A,B,N)
      DIMENSION F(0:M),FA(0:300),FB(0:300)
      DATA PI/3.14159265/
      H=2*PI/M
      TYPE*, H
      J=0
      DO 2 I=0,M
      C=4
      IF (I .EQ. J) C=2
      IF (I .EQ. J) J=J+2
      FA(I)=F(I)*COS(N*H*I)
      FB(I)=F(I)*SIN(N*H*I)
      IF (I .EQ. M) T3=FA(I)
      IF (I .EQ. M) TT3=FB(I)
      IF (I .EQ. M-1) T2=FA(I)
      IF (I .EQ. M-1) TT2=FB(I)
      IF (I .EQ. M-2) T1=FA(I)
      IF (I .EQ. M-2) TT1=FB(I)
      IF (I .EQ. M-3) T0=FA(I)
      IF (I .EQ. M-3) TT0=FB(I)
      IF (I .EQ. 0) GO TO 2
      FA(I)=FA(I-1)+C*FA(I)
      FB(I)=FB(I-1)+C*FB(I)
2     CONTINUE
      A=H/3./PI*(FA(M)-T3)
      B=H/3./PI*(FB(M)-TT3)
      IF (J .NE. M+1) GO TO 3
      A=H/3./PI*(FA(M)-4.*T3-2.*T2-4.*T1-T0)
      A=A+3.*H/8./PI*(T0+3.*T1+3.*T2+T3)
      B=H/3./PI*(FB(M)-4.*TT3-2.*TT2-4.*TT1-TT0)
      B=B+3.*H/8./PI*(TT0+3.*TT1+3.*TT2+TT3)
3     CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CVFIT(NSAMPLES,INFILE,PHASA,UVL,VVL,MENU,MENV,REJ,PREJ,
     1                 MEANU,MEANV,MPHAVG)
C*****************************************************************************
C*                                                                           *
C*  Written by S. D. Hedrick in October, 1987 to fit a sinusoidal curve to LDV *
C*  velocity data generated by an oscillating airfoil.  The subroutine, utiliz- *
C*  ing FOURIER, adapted from June 1977 OCS Airfoil program, fits a single    *
C*  cycle sine wave to the data and computes the u- and v-velocity components *
C*  at a phase angle specified for the analysis.                              *
C*                                                                           *
C*****************************************************************************
      IMPLICIT NONE
      CHARACTER*1 ANS,DUMMY
      CHARACTER*10 INFILE
      INTEGER NSAMPLES,I,PHASA,NSA,REJ,ARG
      REAL FU(0:3600),FV(0:3600),AU,AV,BU,BV,PI,X(3600),Y(3600)
      REAL UVL,VVL,MENU,MENV,PREJ,MEANU(18),MEANV(18),MPHAVG(18)
      DATA PI/3.1415927/
      WRITE(*,*)'DESIRE A VELOCITY/PHASE PLOT OF THE REFINED DATA?'
      WRITE(*,*)'(Y/N--DEF=Y)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') GO TO 7
      CALL RPLOT(NSAMPLES,INFILE,REJ,PREJ)
    7 OPEN(UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 5 I=1,NSAMPLES
        READ(11,100) X(I),Y(I)
    5 CONTINUE
  100 FORMAT(5X,F9.4,1X,F9.4)
      CLOSE(UNIT=11)
      DO 10 I=1,NSAMPLES
        FU(I-1)=X(I)
        FV(I-1)=Y(I)
   10 CONTINUE
      NSA=NSAMPLES-1
      CALL FOURIER(FU,NSA,AU,BU,1)
      CALL FOURIER(FV,NSA,AV,BV,1)
      DO 20 I=0,NSA
        FU(I+1)=MENU+AU*COS((2.*PI/FLOAT(NSAMPLES))*FLOAT(I))+
     1          BU*SIN((2.*PI/FLOAT(NSAMPLES))*FLOAT(I))
        FV(I+1)=MENV+AV*COS((2.*PI/FLOAT(NSAMPLES))*FLOAT(I))+
     1          BV*SIN((2.*PI/FLOAT(NSAMPLES))*FLOAT(I))
   20 CONTINUE
      WRITE(*,*)'PLOTS WILL NOW BE PROVIDED TO COMPARE SINUSOID WITH',
     1          ' VELOCITY DATA.  PRESS <CR> TO CONTINUE.'
      READ(*,'(A1)') DUMMY
      CALL PLOT(NSAMPLES,INFILE,FU,FV,MENU,MENV,MEANU,MEANV,
     1          MPHAVG)
      ARG=NSAMPLES*(FLOAT(PHASA)/360.)
      WRITE(*,*)' ARG =',ARG
      UVL=FU(ARG)
      VVL=FV(ARG)
      RETURN
      END
```

```
      SUBROUTINE PLOT(NSAMPLES,INFILE,FU,FV,MENU,MENV,MEANU,
     1                MEANV,MPHAVG)
****************************************************************************
*                                                                          *
*  Written by S. D. Hedrick in October, 1987 to plot a comparison of velocity *
*  data and a sinusoidal curve fit.                                        *
*                                                                          *
****************************************************************************
      IMPLICIT NONE
      INTEGER  NSAMPLES,I,BOUND,STOPU,STOPV
      REAL*4   XVAL(4000),YVALU(4000),YVALV(4000),XCONTR(4),YCONTU(4)
      REAL*4   YCONTV(4),FU(NSAMPLES),FV(NSAMPLES)
      REAL*4   YMINU,YMAXU,YMINV,YMAXV,MENU,MENV
      REAL     MEANU(18),MEANV(18),MPHAVG(18)
      CHARACTER*10 INFILE
      CHARACTER*11 XLABEL
      CHARACTER*15 YLABEL
      CHARACTER*21 UTITLE
      CHARACTER*19 VTITLE
      CHARACTER*1  DUMMY
      OPEN(UNIT=11,FILE=INFILE,STATUS='OLD')
      DO 10 I=1,NSAMPLES
        READ(11,100) YVALU(I),YVALV(I),XVAL(I)
        XVAL(I)=XVAL(I)/10.
 10   CONTINUE
100   FORMAT(5X,F9.4,1X,F9.4,8X,F4.0)
      XLABEL='PHASE ANGLE'
      YLABEL='VELOCITY (ft/s)'
      UTITLE='HORIZONTAL VELOCITIES'
      VTITLE='VERTICAL VELOCITIES'
      YMINU=MENU-150.
      YMAXU=MENU+150.
      YMINV=MENV-150.
      YMAXV=MENV+150.
      XCONTR(1)=6.0
      XCONTR(2)=0.0
      XCONTR(3)=360.0
      XCONTR(4)=45.0
      YCONTU(1)=6.0
      YCONTU(2)=YMINU
      YCONTU(3)=YMAXU
      YCONTU(4)=50.0
      YCONTV(1)=6.0
      YCONTV(2)=YMINV
      YCONTV(3)=YMAXV
      YCONTV(4)=50.0
      CALL LGP$PLOT(1,'EXSY',XVAL,FU,NSAMPLES,XLABEL,YLABEL, , , ,
     1              XCONTR,YCONTU, ,UTITLE)
      CALL LGP$POINT(1,MPHAVG,MEANU,.03,3,18, , , )
      WRITE(*,*)'PRESS <CR> TO TERMINATE PLOT'
      READ(*,200) DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
      CALL LGP$PLOT(1,'EXSY',XVAL,FV,NSAMPLES,XLABEL,YLABEL, , , ,
     1              XCONTR,YCONTV, ,VTITLE)
      CALL LGP$POINT(1,MPHAVG,MEANV,.03,3,18, , , )
      WRITE(*,*) 'Press <CR> to terminate plot'
      READ(*,200) DUMMY
200   FORMAT(A1)
      CALL LGP$TERMINATE_PLOT(1,1)
      CLOSE(UNIT=11)
      RETURN
      END
```

```
      SUBROUTINE PROPLOT(UNDIM,VNDIM,ZPO,XPOS,N)
      IMPLICIT NONE
      REAL*4 UNDIM(30),VNDIM(30),ZPO(30),XCONTU(4),XCONTV(4)
      REAL*4 YCONTR(4),UNDI(3000),VNDI(3000),ZP(3000)
      REAL XPOS
      INTEGER N,NBR
      CHARACTER*29 XLABEU,XLABEV
      CHARACTER*32 YLABEL
      CHARACTER*27 UTITLE
      CHARACTER*25 VTITLE
      CHARACTER*1  DUMMY,ANS
      CHARACTER*9  XPO
      CHARACTER*4  XXPO
      XLABEU='VELOCITY RATIO (U/U INFIN )'
      XLABEV='VELOCITY RATIO (V/U INFIN )'
      YLABEL='PROBE POSITION (IN. FROM CENTER)'
      UTITLE='HORIZONTAL VELOCITY PROFILE'
      VTITLE='VERTICAL VELOCITY PROFILE'
      XPOS=XPOS/3.0
      WRITE(UNIT=XXPO,FMT='(F4.1)')XPOS
      XPO='X/C = '//XXPO
      XCONTU(1)=6.0
      XCONTU(2)=0.0
      XCONTU(3)=1.2
      XCONTU(4)=0.2
      XCONTV(1)=6.0
      XCONTV(2)=-0.7
      XCONTV(3)=0.7
      XCONTV(4)=0.2
      YCONTR(1)=6.0
      YCONTR(2)=-2.0
      YCONTR(3)=2.0
      YCONTR(4)=0.5
      NBR=10*(N-1)+1
      CALL LGP$PLOT(1,'EXSY',UNDIM,ZPO,0,XLABEU,YLABEL,
     1             , , ,XCONTU,YCONTR, ,UTITLE)
      CALL LGP$POINT(1,UNDIM,ZPO,.03,3,N, , , )
      CALL LGP$PUT_TEXT(1,.4,5.5,XPO, , )
      WRITE(*,*)'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
      CALL LGP$PLOT(1,'EXSY',VNDIM,ZPO,0,XLABEV,YLABEL,
     1             , , ,XCONTV,YCONTR, ,VTITLE)
      CALL LGP$POINT(1,VNDIM,ZPO,.03,3,N, , , )
      CALL LGP$PUT_TEXT(1,.4,5.5,XPO, , )
      WRITE(*,*)'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
      WRITE(*,*)'DO YOU DESIRE A SPLINE INTERPOLATION CURVE FIT OF',
     1          ' THE DATA POINTS? (Y/N)'
      READ(*,'(A1)') ANS
      IF (ANS .EQ. 'N' .OR. ANS .EQ. 'n') GO TO 10
      CALL LGP$PLOT(1,'EXSY',UNDIM,ZPO,0,XLABEU,YLABEL,
     1             , , ,XCONTU,YCONTR, ,UTITLE)
      CALL LGP$POINT(1,UNDIM,ZPO,.03,3,N, , , )
      CALL LGP$SPLINE_QHC(UNDIM,ZPO,N,10,UNDI,ZP,)
      CALL LGP$PLOTM(1,'EXSY',UNDI,ZP,NBR,)
      CALL LGP$PUT_TEXT(1,.4,5.5,XPO, , )
      WRITE(*,*)'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
      CALL LGP$PLOT(1,'EXSY',VNDIM,ZPO,0,XLABEV,YLABEL,
     1             , , ,XCONTV,YCONTR, ,VTITLE)
      CALL LGP$POINT(1,VNDIM,ZPO,.03,3,N, , , )
      CALL LGP$SPLINE_QHC(VNDIM,ZPO,N,10,VNDI,ZP,)
      CALL LGP$PLOTM(1,'EXSY',VNDI,ZP,NBR,)
      CALL LGP$PUT_TEXT(1,.4,5.5,XPO, , )
      WRITE(*,*)'PRESS <CR> TO TERMINATE PLOT'
      READ(*,'(A1)') DUMMY
      CALL LGP$TERMINATE_PLOT(1,1)
10    RETURN
      END
```

LIST OF REFERENCES

1. Young, Warren H., Jr., Fluid Mechanics Mechanisms in the Stall Process of Airfoils for Helicopters, USARTL, NASA Langley, January 1981.

2. McCroskey, W. J., The Phenomenon of Dynamic Stall, NASA Ames, March 1981.

3. NASA Contractor Report 177386, Laser Doppler Velocimetry Primer, by William D. Bachalo, December 1985.

4. Merzkirch, W., Density Sensitive Flow Visualization, Ruhr-Universitat Bochum, Federal Republic of Germany, 1981.

INITIAL DISTRIBUTION LIST

No. Copies

1. Defense Technical Information Center — 2
   Cameron Station
   Alexandria, Virginia 22304-6145

2. Library, Code 0142 — 2
   Naval Postgraduate School
   Monterey, California 93943-5000

3. Satya Bodapati, Code 67 Bu — 10
   Department of Aeronautics
   Naval Postgraduate School
   Monterey, California 93943-5000

4. M. F. Platzer, Chairman — 1
   Department of Aeronautics
   Naval Postgraduate School
   Monterey, California 93943-5000

5. Steven D. Hedrick — 3
   608 Airwood Dr. W.
   E. Alton, Illinois 62024

6. D. P. Bencze — 1
   Chief, Applied Aerodynamics Branch
   NASA Ames Research Center (M. S. 227-6)
   Moffett Field, California 94035

7. Chief, Fluid Mechanics Laboratory — 1
   NASA Ames Research Center (M. S. 260-1)
   Moffett Field, California 94035

8. Engineering Sciences Division — 1
   Attn: T. L. Doligalski,
   Chief, Fluid Dynamics
   U. S. Army Research Office
   P. O. Box 12211
   Research Triangle Park
   North Carolina 27709-2211

9. Dr. M. S. Chandrasekhara — 1
   NASA Ames Research Center (M. S. 260-1)
   Moffett Field, California 94035

10. Directorate of Aerospace Sciences 1
Attn: Hank E. Helin, Program Manager
AFOSR/NA
Bolling Air Force Base
Washington, DC 20332-6448

11. Naval Air Systems Command 1
Attn: Harry Berman, Program Manager
AIR-931K
Washington, DC 20361-9320

12. Office of Naval Research 1
Attn: Spiro Lekoudis, Program Manager
Code 1132F
800 North Quincy Street
Arlington, Virginia 22217-5000

13. David Taylor, NSRDC, Code 8 1
Attn: Jim Nichols
Aviation and Surface Effects Department
Bethesda, Maryland 20084

END
DATE
FILM
4-88
DTIC